… The search for truth, even when it concerns a finite reality of the world or of man, is never-ending, but always points beyond to something higher than the immediate object of study, to the questions which give access to mystery …

John Paul II

University of Alberta

A Fuzzy Computing Framework for Medical Image Registration

by

Lino Martin Ramirez ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering

Edmonton, Alberta
Spring, 2008

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-45586-9
Our file Notre référence
ISBN: 978-0-494-45586-9

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

Medical images are largely used to diagnose medical conditions and to plan, implement, and evaluate surgical and therapeutic procedures. It is therefore crucial that the information obtained from these images be timely, relevant and accurate. Precisely registering images over time or across modalities is necessary before extracting relevant clinical information. The precision of registration solutions is usually verified by visual inspection, which is time consuming. To address this issue, a fuzzy computing framework that could be used to automatically assess the quality of the registration solutions was developed. Fuzzy computing was chosen because of its ability to deal with imprecise or ambiguous regions in images. This work had two objectives: 1) to develop a similarity measure for image registration; and 2) to develop an overlap measure to assess the quality of registration solutions. Since the clinical focus was on spinal deformities, the main interest was on measuring the inclinations of vertebrae in spine radiographs. A registration algorithm based on the proposed similarity measure was applied on 16 vertebral endplates. The similarity measure parameters were adjusted until satisfactory results (as assessed by an expert) were achieved. The registration was successful in all the cases. These results outperformed those obtained using a registration algorithm based on the mean squared errors (MSE) similarity measure (which was successful in 80% of the cases). A registration algorithm based on the proposed similarity measure was applied with fixed registration parameters on 141 vertebral endplates. The proposed overlap measure and image descriptors were then computed. The resulting dataset was divided into training and test set. The training set was used to configure three classifiers: a support vector classifier (SVC), a decision tree classifier (DT) and a logistic regression classifier (LR). Their performance was evaluated on the test set. The SVC had an accuracy of 86% discriminating Large Misregistrations (those above 3°) from Small Misregistrations. This accuracy was better

than that of the LR (76%) and DT (68%). The differentiation between Large Misregistrations and Small Misregistrations using the proposed overlap measure, image data, and a support vector classifier was achieved successfully.

Dedicated to Marlene and Cristian for giving meaning to my life

Acknowledgment

All the praises and thanks to our Lord for giving me the strength and many gifts needed to complete this work and for bringing to my life a number of people that made this work possible. I would like to express my profound gratitude to:

My supervisors, Dr. Nelson Durdle and Mr. Jim Raso for their guidance, support, and encouragement throughout these years of graduate study;

My friends at the Glenrose Rehabilitation Hospital, in particular Mr. Doug Hill and Dr. Edmond Lou for their suggestions and assistance in the development of important parts of my research;

My friends at the Medical Imaging Lab, in particular Ahmed, Peter, and Rami for their continuous encouragement;

The Alberta Provincial CIHR Training Program in Bone and Joint Health for providing me not only financial support but also opportunities to discover how important health-related research is;

Last but not least, to all my friends and family members for their prayers and constant support.

Table of Content

1 Introduction 1

1.1 Objectives 5

1.2 Clinical significance 6

1.3 Engineering significance 6

1.4 Thesis overview 7

2 Literature review 9

2.1 Fuzzy computing 9

2.1.1 Fuzzy sets 10
2.1.2 Logic operations on fuzzy sets 15
2.1.3 Linguistic modifiers 18
2.1.4 Relations between fuzzy sets: Equality and inclusion 20
2.1.5 Comparison operations 20
2.1.6 Fuzzy sets and digital images 24

2.2 Imaging of scoliosis 25

2.2.1 Surface imaging 26
2.2.2 Radiographs 26
2.2.3 Magnetic resonance imaging (MRI) 28
2.2.4 Other investigations 28

2.3 Medical image registration 28

2.3.1 Registration components 30
2.3.2 Registration Process 33

2.4 Image registration of spine images 34

2.5 Assessment of quality of image registration results 35

2.6 Summary 38

3 Combining intensity-based and gradient-based fuzzy Jaccard indices for the design of an image registration similarity measure 39

3.1 Digital images and fuzzy sets 39

3.2 Fuzzy sets similarity measures for medical image registration 40

3.2.1 Fuzzy Jaccard Index on intensity values 43
3.2.2 Fuzzy Jaccard Index on gradient values 45
3.2.3 The *FJI* as a fuzzy sets similarity measure 49

3.3 The Combined Fuzzy Jaccard Index (CFJI) 53

3.3.1 The Combined Fuzzy Jaccard Index (*CFJI*) as a fuzzy sets similarity measure 56
3.3.2 Exploring the sensitivity of the Combined Jaccard Index to various values of α and σ 58

3.4 Study of the performance of the CFJI in single-modality, multimodality, and model-to-image registration problems 66

3.4.1 Single-modality registration 69
3.4.2 Multi-modality registration 70
3.4.3 Model-to-image registration 71

3.5 Results 72

3.5.1 Single-modality registration problem 72
3.5.2 Multi-modality registration problem 73
3.5.3 Model-to-image registration problem 74

3.6 Discussion 76

3.7 Summary 78

4 A support vector machines classifier to assess the quality of registration solutions from fuzzy overlap measures 79

4.1 Introduction 79

4.2 Proposed overlap measure 80

4.3 Proposed use of the system 83

4.4 Patient data set 86

4.5 Support vector classifiers *90*

4.6 Logistic regression *93*

4.7 Decision trees *94*

4.8 Data preparation *95*

4.9 Exploratory data analysis *96*

4.10 Training and testing of the classifiers *105*

4.11 Results *109*

4.12 Discussion *112*

4.13 Summary *115*

5 Discussion 117

5.1 Proposed overlap measure in a similarity measure for image registration *117*

5.2 Proposed overlap measure as a measure of registration quality *120*

5.3 Limitations *123*

6 Conclusion and recommendations for future work 125

6.1 Conclusions *125*

6.2 Recommendations for future work *127*

References 129

APPENDICES 136

Appendix 1: Use of the System 136

Appendix 2: Proposed Similarity Measure 137

Appendix 3: Characterization of the Proposed Similarity Measure 148

Appendix 4: Mean Squared Errors (MSE) Similarity Measure 152

Appendix 5: Model-to-Image Registration 158

Appendix 6: Computation of Measures of Overlap 164

Appendix 7: Open Source Software Contributions................................. 167

List of Tables

Table 3-1: Example of FJI_i computation. 45
Table 3-2: Example of FJI_g computation. 49
Table 3-3: The resulting transformation parameters for registration of Proton Density Magnetic Resonance Images of a brain, using the *CFJI* and *MSE* similarity measures. 72
Table 3-4: The resulting transformation parameters for the registration of a Proton Density Magnetic Resonance Image to a T1 Magnetic Resonance Image of a brain, using the *CFJI* and *MSE* similarity measures. 74
Table 3-5: Registration performance results for the model-to-image registration problem using the *CFJI* and *MSE* similarity measures. 76
Table 4-1: List of features of the training set. 97
Table 4-2: List of features of the test set. 101
Table 4-3: Contingency table used to compare two classifiers 106
Table 4-4: First two-class classification problem: Large Misregistrations versus Small Misregistrations. Test results with the SVC, DT, and LR classifiers 110
Table 4-5: First two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus LR using a 10-fold crossvalidation approach on the training data set. 110
Table 4-6: First two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus DT using a 10-fold crossvalidation approach on the training data set. 110
Table 4-7: Second two-class classification problem: Large Misregistrations versus Small Misregistrations. Test results with the SVC, DT, and LR classifiers on the test set. 111
Table 4-8: Second two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus LR on the test data set. 112
Table 4-9: Second two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus DT on the test data set. 112

List of Figures

Figure 1-1: Cobb angle measurement. 2
Figure 1-2: Diagram of points used to measure the vertebral and inter-vertebral wedging. 3
Figure 1-3: Inter-vertebral wedge angle measurement. 3
Figure 2-1. Example of a membership function representing the concept "*low*" temperature. 13
Figure 2-2. Core, support, and boundaries of a fuzzy set. 13
Figure 2-3. Diagrams for (a) crisp set boundary and (b) fuzzy set boundary. 14
Figure 2-4: Example of a frontal radiograph of a patient with a double scoliotic curve. 27
Figure 2-5: Example of a frontal radiograph of the above spine after having undergone successful surgery. 27
Figure 2-6. Examples of 2D image transformations. 31
Figure 2-7: Functional blocks of a registration process. 34
Figure 3-1: Gradient magnitude images of a lumbar vertebra image. 48
Figure 3-2: Dry-bone radiographs of a thoracic vertebra for the single-modality registration problem. 59
Figure 3-3: Single-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 1.0$. 59
Figure 3-4: Single-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 5.0$. 60
Figure 3-5: Single-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 1.0$. 60
Figure 3-6: Single-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 5.0$. 61
Figure 3-7: Single-modality similarity measure surface plots for $\alpha = 1$ and $\sigma = 1.0$. 61
Figure 3-8: Single-modality similarity measure surface plots for $\alpha = 1.0$ and $\sigma = 5.0$. 62
Figure 3-9: Brain Magnetic Resonance Images (MRI) for the multi-modality registration problem. 63
Figure 3-10: Multi-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 1.0$. 63
Figure 3-11: Multi-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 5.0$. 64
Figure 3-12: Multi-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 1.0$. 64
Figure 3-13: Multi-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 5.0$. 65
Figure 3-14: Multi-modality similarity measure surface plots for $\alpha = 1$ and $\sigma = 1.0$. 65
Figure 3-15: Multi-modality similarity measure surface plots for $\alpha = 1.0$ and $\sigma = 5.0$. 66
Figure 3-16: Reference image (a) and input image (b) for the single-modality registration example. 70
Figure 3-17: Reference image (a) and input image (b) for the multi-modality registration example. 71
Figure 3-18: Sample reference image (a) and model (b) for the model-image registration example. 72
Figure 3-19: Difference before registration (a) and difference after registration using the *CFJI* (b) and *MSE* (c) similarity measures for the single-modality registration example. 73
Figure 3-20: Difference before registration (a) and difference after registration using the *CFJI* (b) and *MSE* (c) similarity measures for the multi-modality registration example. 74
Figure 3-21: Reference image (a), difference before registration (b) and difference after registration (c) for model-to-image registration of a bottom vertebral endplate using the *CFJI* similarity measure. Reference image (d), difference before registration (e) and difference after registration (f) for model-to-image registration of a top vertebral endplate using the *CFJI* similarity measure. 75

Figure 3-22: The four bad registration solutions out of 16 obtained using the *MSE*. 76
Figure 3-23: The registration solutions obtained with the *CFJI* similarity measure that correspond to the four bad registration solutions out of 16 obtained using the *MSE*. 76
Figure 4-1: Graphical user interface used to select vertebral endplates for the measurement of vertebral wedging. 84
Figure 4-2: Graphical user interface used to select vertebral endplates for the measurement of Cobb angle. 84
Figure 4-3: Sample reference image (a) and model (b) for model-to-image registration. 85
Figure 4-4: Measured wedge for vertebra under study. 85
Figure 4-5: Measured Cobb angle. 86
Figure 4-6: Example of a vertebral endplate that satisfied the inclusion criteria. 88
Figure 4-7: Sample reference image used in the model-to-image registration example (a). Difference before registration (b) and difference after registration (c) for a model-to-image registration example of a top vertebral endplate using the *CFJI* similarity measure. 90
Figure 4-8: A decision tree for the classification of three types of flowers: iris-setosa, iris-versicolor, and iris-virginica. 95
Figure 4-9: Scatter plot of *FJIG* versus *MeanG* for the training set. 98
Figure 4-10: Scatter plot of *FJIG* versus *RangeG* for the training set. 98
Figure 4-11: Scatter plot of *FJIG* versus *StDevG* for the training set. 99
Figure 4-12: Scatter plot of *MeanG* versus *RangeG* for the training set. 99
Figure 4-13: Scatter plot of *MeanG* versus *StDevG* for the training set. 100
Figure 4-14: Scatter plot of *RangeG* versus *StDevG* for the training set. 100
Figure 4-15: Scatter plot of *FJIG* versus *MeanG* for the test set. 102
Figure 4-16: Scatter plot of *FJIG* versus *RangeG* for the test set. 102
Figure 4-17: Scatter plot of *FJIG* versus *StDevG* for the test set. 103
Figure 4-18: Scatter plot of *MeanG* versus *RangeG* for the test set. 103
Figure 4-19: Scatter plot of *MeanG* versus *StDevG* for the test set. 104
Figure 4-20: Scatter plot of *RangeG* versus *StDevG* for the test set. 104

List of Abbreviations and Symbols

FJI_g	Fuzzy Jaccard Index Similarity Measure on Gradient Magnitudes
FJI_i	Fuzzy Jaccard Index Similarity Measure on Intensity Values
CFJI	Combined Fuzzy Jaccard Index Similarity Measure
α	Weighting Factor of the *CFJI*
σ	Smoothing Factor of the *CFJI*
min	Minimum of two values
max	Maximum of two values
t	t-norm
s	s-norm
MSE	Mean Squared Error Similarity Measure
SM	Similarity Measure
r	Pearson Correlation Coefficient
r^2	Coefficient of Determination
FJIG	Fuzzy Jaccard Index Overlap Measure on Gradient magnitudes
MeanG	Mean Value Image Descriptor on Gradient Magnitudes
MedianG	Median Value Image Descriptor on Gradient Magnitudes
RangeG	Range of Values Image Descriptor on Gradient Magnitudes
StDevG	Standard Deviation Image Descriptor on Gradient Magnitudes
SVC	Support Vector Classifier
DT	Decision Tree Classifier
LR	Logistic Regression Classifier
LM	Large Misregistration
SM	Small Misregistration
SN	Sensitivity
SP	Specificity
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
ROI	Region of Interest
MRI	Magnetic Resonance Imaging

1 Introduction

Scoliosis [58], [40] is a condition that involves abnormal lateral curvature and rotation of the spine and usually causes noticeable trunk distortions. These distortions may include asymmetrical elevations of the shoulders and/or hips, prominence of a scapula, and subtle twisting of the trunk. Scoliosis affects between 2 and 4% of adolescents [58] and between 70 and 80% of the cases have an unknown cause [40]. The most common descriptor of the severity of scoliosis is the Cobb angle [13]. This radiographic indicator is obtained from standing posterior-anterior (PA) radiographs of the spine. The Cobb angle is measured between the inflection points of the spine curve (see Figure 1-1). Two other radiographic indicators of interest in spinal deformity research are the vertebral wedge angle and the inter-vertebral wedge angle [2], [66], [1]. Wedge angles are determined by assessing the vertebral endplates inclinations [2]. The inclination of the upper vertebral endplate is given by the inclination (with respect to the horizontal axis) of the line going through points 1 and 2 in Figure 1-2. The inclination of the lower vertebral endplate is given by the inclination (with respect to the horizontal axis) of the line going through points 3 and 4 in Figure 1-2. The vertebral wedge angle is the difference in angle between the inclinations of the upper and lower vertebral endplates. The inter-vertebral wedge angle is the difference in angle between the inclination of the lower endplate of the superior vertebra and the inclination of the upper endplate of the inferior vertebra. For example, the inter-vertebral wedge angle between vertebra A and vertebra B in Figure 1-3 would be equal to the difference of the inclinations of the line going through points 3 and 4 of vertebra A and the inclination of the line going through points 1 and 2 of vertebra B. The measurement of the wedge and Cobb angles are subject to technical errors [66]. In the case of the Cobb angle, errors are associated with the vertebral endplate selection and with intrinsic intra- and inter-observer measurements variability [46]. Measurements of the Cobb angle have an associated intra- and inter-observer variability of approximately 4° to 8° [23]. In the case of wedging, vertebra selection does not contribute to measurement error [66]. Therefore, the error in wedging measurement is expected to be approximately 4°, based on the study of Morrissy et al. [46]. Other errors result from variations in patient posture, and these would affect the measures of Cobb angle and inter-vertebral wedging but not of vertebral wedging [66]. In addition to the errors associated with the measurement of wedging, it is important to note that measuring vertebral and inter-vertebral wedging is a time consuming endeavour. Therefore, the development of a system that assists clinicians as an accurate and reliable screening tool to assess and monitor changes in vertebral morphology and scoliotic curve severity is justified.

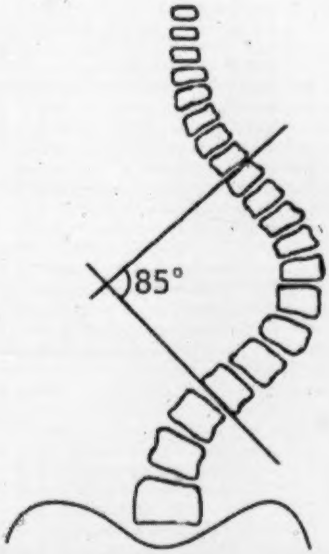


Figure 1-1: Cobb angle measurement. The Cobb angle is the difference in angle between lines going through the upper and lower endplates of the vertebrae most tilted into the concavity of the curve. In the case of the image above, the Cobb angle is equal to 85°.

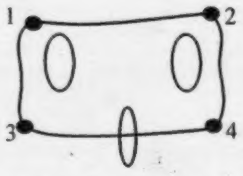


Figure 1-2: Diagram of points used to measure the vertebral and inter-vertebral wedging.

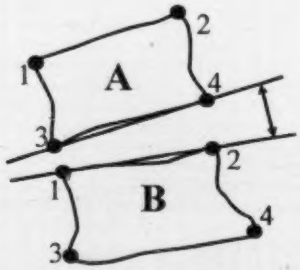


Figure 1-3: Inter-vertebral wedge angle measurement. The inter-vertebral wedge angle is the difference in angle between the line going through the lower endplate of the superior vertebra and the line going through the upper endplate of the inferior vertebra.

Measurement of Cobb and wedge angles involves the identification of vertebral endplates in radiographs, the matching of the endplates with a pre-defined anatomical model (a line), and the establishment of relations between models (i.e., measuring the angles). This task can be automated with the help of image registration. Image registration [24], [27], [84] finds a geometric relation or transformation between corresponding anatomical structures in two images or between a model and an anatomical structure in an image. The main components of any image registration algorithm are geometrical transformation, similarity measure, interpolation method, and optimization strategy. Geometrical transformations align corresponding objects in two or more images. Similarity measures quantify the quality of the match of the images. Interpolation methods estimate pixel intensity values at requested positions resulting from transforming points from one image to another. Optimization strategy refers to the iterative approach of adjusting the transformation parameters in an attempt to improve the similarity measure.

Spine radiographs of patients with scoliosis present two main challenges for image registration: first, they have variable contrast and second, they have moderate focus. To deal with these challenges, a computing framework that can deal with imprecision is needed. One of such frameworks is fuzzy computing. Fuzzy computing [29] is concerned with computing using fuzzy sets. A fuzzy set [82], [83], [50] is a collection of objects with associated membership values on the real continuous interval [0,1]. Because fuzzy sets are able to express the notion of partial membership of an element to a particular group, they can handle uncertainty and imprecision. This property has been instrumental in the success of fuzzy computing in applications that range from image processing and pattern recognition to decision support systems and fuzzy control systems. Fuzzy computing was chosen for the proposed image registration application because it provides a suitable methodology for analyzing complex systems when the pattern indeterminacy is due to inherent variability or fuzziness as is the case in image processing applications. Moreover, fuzzy computing has been used to incorporate knowledge in the determination of the transformation and correspondence and to select and pre-process the features to be registered [55].

Before using the results obtained with medical image registration applications, it is necessary to verify the goodness of the alignment because any error or uncertainty in the registration may lead to uncertainties in diagnosis or in planning, implementing, and evaluating surgical and therapeutic procedures. Assessment of registration accuracy is usually done by visual inspection. Unfortunately, this approach is time consuming and highly dependant on the level of expertise of the radiologist, physician, or other clinical expert. This complicates the adoption of image registration methods in clinical practice. To address this issue, a classification strategy that would indicate whether the registration was successful or not was developed. An innovative approach to develop such a classifier strategy was a challenging task because it required the incorporation of both image information and domain knowledge to successfully distinguish good registrations solutions from bad registration solutions. To deal with these challenges a fuzzy computing framework was used to get an estimation of the overlap between corresponding anatomical structures in the registered images.

Support Vector Classifiers (SVC) [76] were used to find a relation between the proposed measure of fuzzy overlap and registration quality. The main goal was to determine whether SVC, using the proposed measure of fuzzy overlap, could predict registration quality sufficiently well to be used in clinical practice. SVC were chosen because they usually outperform techniques such as Artificial Neural Networks when trained using small datasets as is usually the case in scoliosis research. Moreover, unpublished preliminary studies comparing the performance of radial basis function neural networks [3] and SVC indicated the superiority of the latter in our datasets of scoliosis patients. Finally, the results of applying a SVC to the dataset of scoliosis patients were compared to those obtained by applying logistic regression [65] and classification decision trees [12] to the dataset.

1.1 Objectives

This work had two objectives: a clinical objective and an engineering objective. The clinical objective was to develop a system for measuring inclinations of vertebrae in spine radiographs of patients with scoliosis. The engineering objective was to develop a novel overlap measure that could be used to assess the quality of medical image registration results. The engineering objective involved two main tasks:

1. to study the proposed overlap measure to control image registration processes; and
2. to use the overlap measure to assess the quality of registration results in a model-to-image registration application.

1.2 Clinical significance

This work developed a new research tool to assess and monitor changes in vertebral morphology and changes in scoliotic curve severity. Scoliosis is a condition that affects between 2 and 4% of adolescents [58]. It involves lateral deviation and rotation of the spine. Depending on the severity of the scoliotic curve, the patient could remain in observation, could be a candidate for bracing or may be a surgical candidate [7]. The underlying cause of most scoliosis cases remains unknown [40]. The techniques developed from this research promise to provide a system that will assist clinicians as an accurate and reliable screening tool in assessing vertebral and inter-vertebral wedging and assessing the severity of the scoliotic curve. Moreover, the proposed system will provide clinicians with a quantitative profile of the changes associated with the scoliotic process, which is crucial for studying their association with suspected risk factors and symptoms.

1.3 Engineering significance

This work developed an intelligent system for automatically evaluating medical image registration results. Such a system includes the following components:

1. The identification of a feature space for image registration that facilitates the incorporation of spatial information into intensity-based image registration approaches.
2. The development of a theoretical basis for the evaluation of a similarity measure based on the new feature space.
3. The development of a theoretical basis for and evaluation of a reliable measure of the overlap between registered images.
4. The development and evaluation of a reliable indicator of registration quality.
5. The testing of the proposed framework on a clinical application to measure vertebrae inclinations.

The techniques developed from this research promise to provide the foundations for the application of fuzzy computing for medical image registration. Moreover, the excellent results obtained with the proposed framework may encourage the development of new intelligent systems based on fuzzy computing and support vector classifiers to automatically assess the quality of registration results.

1.4 Thesis overview

This thesis consists of six chapters. Chapter 2 reviews the literature on imaging of scoliosis, fuzzy computing, medical image registration, the application of image registration in the analysis of spine images, and the assessment of goodness of fit in image registration applications. Chapter 3 describes the proposed similarity measure for medical image registration in a fuzzy computing framework. A registration algorithm based on the proposed similarity measure is used for model-to-image registration of vertebral endplates in spine radiographs. Additional numerical simulations are presented to illustrate the applicability of the proposed similarity measure in other medical image registration applications. Chapter 4 introduces the proposed methodology to assess the goodness of the image registration solution. Detailed numerical studies, involving the model-to-image registration of vertebral endplates, are presented to determine whether the proposed system could assess the quality of the registration solution sufficiently well to be used in clinical practice. Chapter 5 is devoted to the discussion of results obtained in the course of the research work. Chapter 6 contains the conclusions resulting from this research, the summary of contributions, and a series of recommendations for future work.

The appendices contain a description of the proposed use for this work's contributions, the algorithms and source code associated with this work, and a list of software contributions made to the premier open source image analysis project, the Insight Segmentation and Registration Toolkit (ITK, www.itk.org) [33].

2 Literature review

This chapter introduces the background on fuzzy computing [29]. The main focus is on the operations available for computing with fuzzy sets [82], [83], [50]. Next, a review of the literature on imaging of scoliosis is presented. After that, the background on medical image registration [24] is presented. It includes the main application areas of a medical image registration solution, the main components of an image registration solution, and a description of the image registration process. Also, this chapter reviews the literature on the application of image registration for the quantitative analysis of spine images. The chapter concludes with a review of the literature on the assessment of the goodness of fit of registration solutions

2.1 Fuzzy computing

It is human nature to use abstractions to deal with the large amount of information that surround them. These abstractions can be represented by sets. Once a set is defined, objects are classified either as members or not of the set. This type of binary classification works well in a number of straightforward cases such as the set of "even numbers", the set of "odd numbers", and the set of "Canadian provinces". Binary classification does not work as well when there is a need to express notions such as "tall people", "heavy snowfall" and "low temperature". To properly understand those notions, there is a need to have some context associated with them. For instance, "low temperature" in Edmonton, Alberta could mean a temperature of around -20°C in winter time or a temperature of around 5°C in summer time. Moreover, a "low temperature" in Caracas, Venezuela could mean a temperature of around 15°C all year around. As the previous examples show, time and location play a role in defining the notion "low temperature". In addition to time and location, personal preferences and previous experiences affect the definition of "low temperature". To deal with notions such as "low temperature", Zadeh introduced the concept of fuzzy sets [82]. A fuzzy set [82], [83], [50] is a collection of objects with associated membership values on the real continuous interval [0,1], where the endpoints 0 and 1 represent no membership to the collection and full membership to the collection respectively. The contexts associated with the everyday life notions ("low temperature", "heavy snowfall", "tall people", etc.) are incorporated with the help of membership functions (which map objects from the problem domain to the unit interval [0,1]). Fuzzy computing [29] is concerned with computing using fuzzy sets. Fuzzy computing has been successfully used in applications that range from image processing and pattern recognition to decision support systems and fuzzy control systems [63]. The rest of this section will cover in more detail the theory of fuzzy sets and its applicability in image processing.

2.1.1 Fuzzy sets

A fuzzy set [50],[82] A in a universe of discourse $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$ is defined by a mapping from the universe of discourse to the interval [0,1], i.e.:

$$A : \mathbf{X} \to [0,1] \qquad (2\text{-}1)$$

Thus, a fuzzy set A in $\mathbf{X}$ may be represented as a set of ordered pairs $\{(x_1, A(x_1)), (x_2, A(x_2)), \ldots, (x_n, A(x_n))\}$, where $A(x_i)$ describes a degree of membership of x_i in A. Consider, for example, the concept of *low* temperature in an environmental context with temperatures distributed in the interval [0,100] defined in degrees Celsius (°C). Clearly, 100°C will not represent a *low* temperature in that interval, and one may assign it a membership degree of zero to the concept of *low* temperature (i.e., $L(100)=0$, with L being the fuzzy set of *low* temperature). Similarly, 10°C and below are certainly *low* temperatures, and one may assign a membership degree of one to express a full degree of compatibility with the concept of *low* temperature (i.e., $L(10)=1, L(9)=1, L(8)=1$, …, $L(0)=1$). The degree of belongingness for the remaining temperature values can be determined through a mapping such as the one depicted in Figure 2-1, which is a membership function $L:\mathbf{T}\rightarrow[0,1]$ characterizing the fuzzy set L of low temperatures in the universe **T**=[0,100].

As fuzzy sets are described by membership functions, it is useful to develop a lexicon of terms to describe various special features of these functions. Figure 2-2 assist in this description.

- Core [50], [62]. The core of a membership function for some fuzzy set A, in a universe **X**, is the set of all elements of **X** that have a full membership in the fuzzy set A. That is

$$Core(A)=\{x\in\mathbf{X}/A(x)=1\} \tag{2-2}$$

- Support [50], [62]. The support of a membership function for some fuzzy set A, in a universe **X**, is the set of all elements of **X** that have a nonzero membership in the fuzzy set A. That is

$$Supp(A)=\{x\in\mathbf{X}/A(x)>0\} \tag{2-3}$$

- Boundaries [50], [62]. The boundaries of a membership function for some fuzzy set A, in a universe **X**, is the set of all elements of **X** that have a nonzero membership but not complete membership in the fuzzy set A. That is

$$Bound(A) = \{x \in \mathbf{X} / 0 < A(x) < 1\} \qquad (2\text{-}4)$$

- Height [50], [62]. The height of a membership function for some fuzzy set A, in a universe **X**, is the maximum value of the membership function. That is

$$hgt(A) = \sup_x \{A(x)\} \qquad (2\text{-}5)$$

- Normality [50], [62]. A normal fuzzy set is one whose height is equal to 1.

- λ-cut [50], [62]. An λ-cut of a membership function for some fuzzy set A, in a universe **X**, is given by

$$A_\lambda = \{\, x \in \mathbf{X} / A(x) \geq \lambda \,\} \qquad (2\text{-}6)$$

- Cardinality [50], [62]. The cardinality of a membership function for some fuzzy set A, in a universe **X**, is given by

$$card(A) = \sum_{x \in \mathbf{X}} A(x) \qquad (2\text{-}7)$$

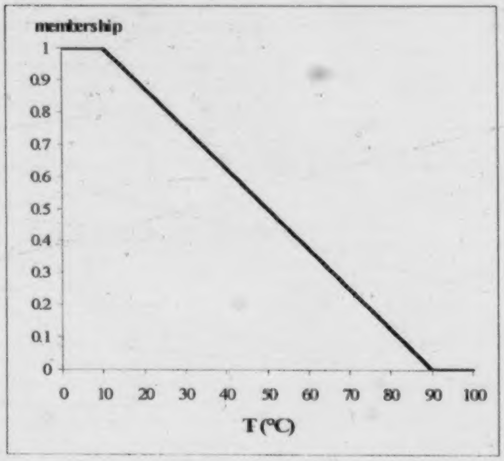


Figure 2-1. Example of a membership function representing the concept "*low*" temperature.

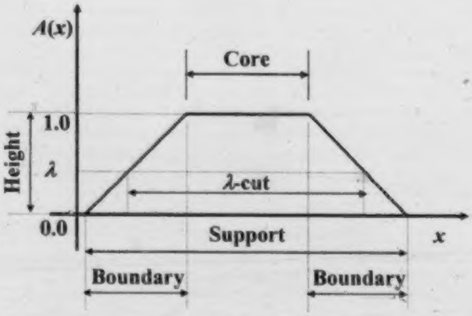
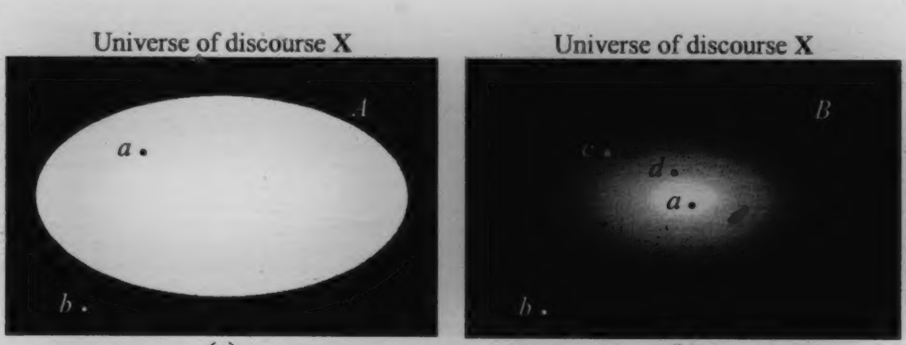


Figure 2-2. Core, support, and boundaries of a fuzzy set.

To illustrate the concepts of crisp set and fuzzy set memberships from a two-dimensional perspective, one can consider the example illustrated in Figure 2-3. For a given universe of discourse **X**, Figure 2-3a shows a crisp set *A* in which the borders are clearly defined. In Figure 2-3a, point *a* is a member of crisp set *A* while point *b* is unambiguously not a member of set *A*. Figure 2-3b shows the vague, ambiguous boundary of a fuzzy set *B* on the same universe of discourse **X**. The shaded boundary represents the boundary of *B*. In the central (completely white) area of the fuzzy set, point *a* is a full member of the fuzzy set.

Outside the boundary region, point *b* is not a member of the fuzzy set. The membership of points *c* and *d*, which are in the boundary region, is ambiguous. If complete membership to the fuzzy set (such as point *a* in Figure 2-3b) is represented by the number 1 and no membership to the fuzzy set (such as point *b* in Figure 2-3b) is represented by the number 0, then points *c* and *d* in Figure 2-3b must have some intermediate value of membership (partial membership to the fuzzy set). Presumably, the point *d* in Figure 2-3b has a membership value greater than the membership value of the point *c* because it is closer to the central (completely white) area of the fuzzy set *B*.

Crisp sets are nondifferentiable constructs. Their usage reduces the applicability of gradient-based optimization tools. As a consequence, when working with crisp sets, one usually resorts to optimization tools that can provide global optimization and do not require derivative information (such as random search or genetic algorithms [22]). In contrast, fuzzy sets are differentiable constructs. Therefore, when working with fuzzy sets, one can use gradient-based optimization tools [29].



(a) (b)

Figure 2-3. Diagrams for (a) crisp set boundary and (b) fuzzy set boundary.

Fuzzy sets can be regarded as linguistic granules. Linguistic granules can be common sense notions (tall individuals, high inflation, low interest rate, etc.) or specific technical terms (high power consumption, small positive error in a control loop, ill-defined matrix, etc.). Terms such as high oil royalty rate and low employee turnover are highly descriptive and meaningful. These terms are often more useful than precise descriptions such as oil royalty rate of 20.25% or 2.25% employee turnover. By using fuzzy sets, one can summarize the domain knowledge in a convenient format. On the operational side of the technology of fuzzy sets, there is a large group of methods that supports all facets of computing with fuzzy sets. Logic operations on fuzzy sets, linguistic modifiers, and comparison operations are a few among the operations that are available.

2.1.2 Logic operations on fuzzy sets

The AND, OR, and NOT operators of boolean logic exist in fuzzy logic [83]. The AND operator is used to find the intersection between two sets (it was originally defined as the minimum between two Fuzzy Sets [83]). The OR operator is used to find the union between two sets (it was originally defined as the maximum between two Fuzzy Sets [83]). The NOT operator is used to find the complement of a set (it was originally defined as the subtraction of a membership function from 1 [83]). The intersection and the union are completed via **t**-norms and **s**-norms [50].

$$A \cap B : \left(A(x) \cap B(x)\right) = A(x)\,\mathbf{t}\,B(y) \tag{2-8}$$

$$A \cup B : \left(A(x) \cup B(x)\right) = A(x)\,\mathbf{s}\,B(y) \tag{2-9}$$

$$\overline{A} : \overline{A}(x) = 1 - A(x) \qquad (2\text{-}10)$$

A **t**-norm is a two argument function $\mathbf{t} : [0,1]^2 \rightarrow [0,1]$ that satisfies the following conditions [50]:

- Commutativity: $x \,\mathbf{t}\, y = y \,\mathbf{t}\, x$
- Associativity: $x \,\mathbf{t}\, (y \,\mathbf{t}\, z) = (x \,\mathbf{t}\, y) \,\mathbf{t}\, z$
- Monotonicity: If $x \leq y$ and $w \leq z$, then $x \,\mathbf{t}\, w \leq y \,\mathbf{t}\, z$
- Boundary condition: $0 \,\mathbf{t}\, x = 0,\ 1 \,\mathbf{t}\, x = x$

An **s**-norm is a two argument function $\mathbf{s} : [0,1]^2 \rightarrow [0,1]$ that satisfies the following conditions [50]:

- Commutativity: $x \,\mathbf{s}\, y = y \,\mathbf{s}\, x$
- Associativity: $x \,\mathbf{s}\, (y \,\mathbf{s}\, z) = (x \,\mathbf{s}\, y) \,\mathbf{s}\, z$
- Monotonicity: If $x \leq y$ and $w \leq z$, then $x \,\mathbf{s}\, w \leq y \,\mathbf{s}\, z$
- Boundary condition: $0 \,\mathbf{s}\, x = x,\ 1 \,\mathbf{s}\, x = 1$

Some commonly used **t**-norms are:

$$x \,\mathbf{t}\, y = min(x, y) \tag{2-11}$$

$$x \,\mathbf{t}\, y = x \cdot y \tag{2-12}$$

$$x \,\mathbf{t}\, y = max(0, (1 + p) \cdot (x + y - 1) - p \cdot x \cdot y), \quad p \geq -1 \tag{2-13}$$

Some commonly used **s**-norms are:

$$x \,\mathbf{s}\, y = max(x, y) \tag{2-14}$$

$$x \,\mathbf{s}\, y = x + y - x \cdot y \tag{2-15}$$

$$x \,\mathbf{s}\, y = min(1, x + y + p \cdot x \cdot y), \quad p \geq 0 \tag{2-16}$$

For a comprehensive list of examples of **t**-norms and **s**-norms, the reader is referred to [50].

2.1.3 Linguistic modifiers

Linguistic modifiers or linguistic hedges are operators that modify linguistic terms from their original interpretation. Examples of linguistic modifiers are *very*, *plus*, *slightly*, and *minus* [62]. For a given fuzzy set A, one can define those linguistic modifiers as follows:

$$very(A) = A^{2}(x) \tag{2-17}$$

$$plus(A) = A^{1.25}(x) \tag{2-18}$$

$$slightly(A) = A^{0.5}(x) \tag{2-19}$$

$$minus(A) = A^{0.75}(x) \tag{2-20}$$

Very and *plus* [62] are linguistic modifiers known as concentrations. Concentration reduces the membership values of a fuzzy set concentrating them around points with a higher membership grade. For instance, by using equation (2-17) for the linguistic modifier *very*, a membership value of 0.9 is reduced by 10% to a value of 0.81 while a membership value of 0.1 is reduced by 90% to a value of 0.01. This decrease is a result of the properties of membership values: for $0 \leq A(x) \leq 1$, then $A^{2}(x) \leq A(x)$. Concentrations can be generalized by using any exponent $p>1$, i.e.:

$$Con(A) = A^{p}(x) \tag{2-21}$$

Slightly and *minus* [62] are linguistic modifiers known as dilations. Dilation increases the membership value of a fuzzy set stretching them around points with lower membership grade. For instance, by using equation (2-19) for the linguistic modifier slightly, a membership value of 0.9 is increased 5.4% to a value of 0.95 while a membership value of 0.1 is increased by 216% to a value of 0.32. This increase is a result of the properties of membership values: for $0 \le A(x) \le 1$, then $A^{0.5}(x) \ge A(x)$. Dilations can be generalized by admitting any exponent $r \in (0,1)$, i.e.:

$$Dil(A) = A^r(x) \tag{2-22}$$

Concentrations and dilations can be combined to produce a plethora of new linguistic modifiers. Two such modifiers are contrast intensification and fuzzification [62]. Contrast intensification increases the membership values of those elements with original membership values greater than 0.5 and it decreases the membership values of those elements with original membership values less than 0.5. The operation is defined by

$$Int(A) = \begin{cases} 2 \cdot A^2(x), & \text{if } 0 \le A(x) \le 0.5 \\ 1 - 2 \cdot (1 - A(x))^2, & \text{otherwise} \end{cases} \tag{2-23}$$

Contrast intensification can be generalized by using any exponent $p>1$, i.e.:

$$Int(A) = \begin{cases} 2^{p-1} \cdot A^p(x), & \text{if } 0 \le A(x) \le 0.5 \\ 1 - 2^{p-1} \cdot (1 - A(x))^p, & \text{otherwise} \end{cases} \tag{2-24}$$

Fuzzification is complementary to contrast intensification. It is produced by altering the original membership function with the following expression

$$Fuzz(A)=\begin{cases}\sqrt{A(x)/2}, & \text{if } 0\le A(x)\le 0.5\\ 1-\sqrt{(1-A(x))/2}, & \text{otherwise}\end{cases} \qquad (2\text{-}25)$$

2.1.4 Relations between fuzzy sets: Equality and inclusion

Equality. Two fuzzy sets A and B, defined in the same universe of discourse **X**, are said to be equal if and only if (iff) their membership functions are identical [50]. That is:

$$A = B\text{, iff } A(x)=B(x), \forall x \in \mathbf{X}$$

Inclusion. Given two fuzzy sets A and B, defined in the same universe of discourse **X**, A is said to be included in B if and only if (iff) the membership function of A is less than or equal to that of B for each x in **X** [50]. That is:

$$A \subseteq B\text{, iff } A(x)\le B(x), \forall x \in \mathbf{X}$$

2.1.5 Comparison operations

Given two fuzzy sets A and B, defined in the same universe of discourse **X**, there are a plethora of measures proposed to express the extent to which the two fuzzy sets match. Several classes of methods available today for satisfying this objective are presented. The main classes of methods are reviewed and the aspects of matching supported by them are highlighted.

- **Possibility measure**. The possibility measure, denoted by *Poss*(*A*, *B*), describes a level of overlap between two fuzzy sets [50]. It is defined as

$$Poss(A,B) = \max_{x \in \mathbf{X}}\left[A(x)\ \mathbf{t}\ B(x)\right] \tag{2-26}$$

Computationally, the possibility measure is concerned with the determination of the intersection between *A* and *B*, $A(x)$ **t** $B(x)$, followed by the optimistic assessment of this intersection. This is achieved by selecting the highest values among the intersection grades of *A* and *B* that are taken over all elements of the universe of discourse **X**. The possibility measure is symmetric: $Poss(A,B) = Poss(B,A)$.

- **Necessity measure**. The necessity measure, denoted by *Nec*(*A*, *B*), describes a degree to which *B* is included in *A* [50]. It is defined as

$$Nec(A,B) = \min_{x \in \mathbf{X}}\left[(1 - A(x))\ \mathbf{s}\ B(x)\right] \tag{2-27}$$

Computationally, the necessity measure is concerned with the determination of the union between (1-*A*) and *B*, (1-*A*(*x*)) **s** *B*(*x*), followed by the pessimistic assessment of this union. This is achieved by selecting the lowest values among the union grades of (1-*A*) and *B* that are taken over all elements of the universe of discourse **X**. The necessity measure is asymmetrical: $Nec(A,B) \neq Nec(B,A)$.

The next two types of operations do not place the comparison procedure in the set theoretic perspective.

- **Distance measures**. Distance measures view membership functions as real functions and compute the distance between *A* and *B*. Because the computations involve two functions, they emphasize the functional facet of fuzzy sets [50]. The more similar the two fuzzy sets, the lower the distance function between them. In general, the distance between *A* and *B* can be defined using the Minkowski distance

$$d(A,B)=\left[\sum_{x\in\mathbf{X}}|A(x)-B(x)|^{p}\right]^{1/p} \tag{2-28}$$

where $p\geq 1$. Depending on the value of *p*, one can distinguish several well-known forms of the distance function [12].

Hamming distance (p=1):

$$d(A,B)=\sum_{x\in\mathbf{X}}|A(x)-B(x)| \tag{2-29}$$

Euclidean distance (p=2):

$$d(A,B)=\left[\sum_{x\in\mathbf{X}}|A(x)-B(x)|^{2}\right]^{1/2} \tag{2-30}$$

Tchebyschev distance ($p\rightarrow\infty$):

$$d(A,B)=\max_{x\in\mathbf{X}}|A(x)-B(x)| \tag{2-31}$$

The distance function $d(A,B)$ satisfies these requirements:

i. $d(A,B) = 0$ if and only if $A=B$

ii. $d(A,B) \geq 0$ (nonnegative function)

iii. $d(A,B) = d(B,A)$ (symmetry)

iv. $d(A,C) \leq d(A,B) + d(B,C)$ (triangle inequality)

- **Similarity measures**. Similarity measures [9], [18], [78], denoted by $S(A, B)$, describe a degree to which B resembles A. In this thesis, a fuzzy similarity is defined in this way:

Let $F(x)$ be the class of all fuzzy sets of $\mathbf{X}$ and let S be a mapping $S : F(x) \times F(x) \rightarrow [0,1]$

Definition: $S(A,B)$ is said to be a degree of similarity between $A \in F(x)$ and $B \in F(x)$ if $S(A,B)$ satisfies the following properties:

- P1 (Boundary Conditions): $0 \leq S(A,B) \leq 1$
- P2: (Reflexivity): if $A = B$, $S(A,B) = 1$
- P3 (Commutativity): $S(A,B) = S(B,A)$
- P4 (Transitivity): if $A \subseteq B \subseteq C$ $A,B,C \in F(x)$ then $S(A,C) \leq S(A,B)$ and $S(A,C) \leq S(B,C)$

Four examples of similarity measures are [16]:

$$S_1(A,B) = 1 - \frac{\sum_{x \in X} |A(x) - B(x)|}{\sum_{x \in X} |A(x) + B(x)|} \quad (2\text{-}32)$$

$$S_2(A,B) = \frac{card(A \cap B)}{card(A \cup B)} \quad (2\text{-}33)$$

$$S_3(A,B) = \frac{card(A \cap B)}{max(card(A), card(B))} \quad (2\text{-}34)$$

$$S_4(A,B) = \frac{card(A \cap B)}{min(card(A), card(B))} \quad (2\text{-}35)$$

2.1.6 Fuzzy sets and digital images

The growing interest of fuzzy sets for image analysis is due to their ability to represent imprecise, uncertain or ambiguous regions or structures in images [15], [4], [5], [6], [61]. Imprecision in medical images can be found at all levels from the observed phenomenon itself to image processing results. Consider, for example, tumour segmentation. A tumour is often difficult to segment precisely due to the smooth transition between healthy and pathological tissues. This is an example of imprecision due to the observed phenomenon. This type of imprecision will be present whatever the choice of imaging technique. Imprecision at the sensor level is usually caused by coarse resolution of the sensors. Imprecision in the image processing results is caused by imprecise image processing operators [4].

A digital image (having various shades of grey or intensity values) can be identified with a fuzzy set A that takes values on the grid points (i, j). Each value denotes the degree of brightness level or some relative pixel density depending on the problem being solved [16]. The brightness levels might vary from a state of no brightness, or completely black, to a state of complete brightness, or completely white. Brightness levels in between these two extremes would get increasingly lighter as they move from black to white. Using the notation of fuzzy sets, we can write the fuzzy set representation of a digital image as an *M* x *N* array,

$$A = \begin{bmatrix} A(x_{11}) & A(x_{12}) & \cdots & A(x_{1j}) & \cdots & A(x_{1N}) \\ A(x_{21}) & A(x_{22}) & \cdots & A(x_{2j}) & \cdots & A(x_{2N}) \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ A(x_{i1}) & A(x_{i2}) & \cdots & A(x_{ij}) & \cdots & A(x_{iN}) \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ A(x_{M1}) & A(x_{M2}) & \cdots & A(x_{Mj}) & \cdots & A(x_{MN}) \end{bmatrix} \qquad (2\text{-}36)$$

where $i, j \in \aleph$, $0 \le i \le M$ and $0 \le j \le N$ with *M* and *N* being the dimensions of the digital image. $A(x_{ij})$ $(0 \le A(x_{ij}) \le 1)$ represents the membership value of the (*i,j*)th pixel to some property *A*. The fuzzy property $A(x_{ij})$ may be defined in a number of ways with respect to brightness level, gradient magnitude, or other image property depending on the problem. Once $A(x_{ij})$ is defined, all the operations on fuzzy sets can be applied to the digital images. Therefore, it is possible to use fuzzy sets to deal with the imprecision, uncertainty, and ambiguity characteristics of images at different levels of an image analysis application.

2.2 Imaging of scoliosis

For treating patients with scoliosis, orthopaedic surgeons need diagnostic imaging techniques that provide information on the possible underlying disease and prognosis to choose the best possible treatment [70]. After treatment is instituted, orthopaedic surgeons use imaging for monitoring changes of the deformity and to optimize therapy [70]. The main types of imaging techniques used in the investigation and management of scoliosis are surface imaging, radiographs, and magnetic resonance imaging.

2.2.1 Surface imaging

Several methods have been introduced to find relations between trunk surface and spine deformity. Most of them have focused on the analysis of back surface images (e.g., [56], [28], [69], [21], [43], [67], [72]) obtained using 1 laser scanner or projection of patterns. Recently, 360° images of the torso (obtained using 4 laser scanners) have also been used [36], [37]. The full torso scan systems provide more information about the 3-D nature of the deformity than back surface-based methods. However, full torso scan systems have not been widely used clinically because they may provide only a small increase in the correlation to the Cobb angle and cost substantially more than back surface scan systems.

2.2.2 Radiographs

Standing films of the whole spine, including parts of the pelvis, are the gold standard of scoliosis diagnosis, management, and pre- and post- operative assessment [70], [7]. At the initial visit, plain films of the whole spine in the frontal and lateral views have to be taken. In successive visits, a single frontal film is often sufficient [7]. The severity of the curve and its progression over time is measured using Cobb's method on the frontal radiograph [70], [7]. Figure 2-4 and Figure 2-5 show frontal radiographs of the same patient. Figure 2-4 shows a frontal radiograph taken before surgery. The scoliosis curve depicted is a double curve with a right thoracic curve having a Cobb angle of 30° and a left lumbar curve having a Cobb angle of 53°. Figure 2-5 shows a frontal radiograph taken after surgery. The Cobb angles of the thoracic and the lumbar curves are below 15°.

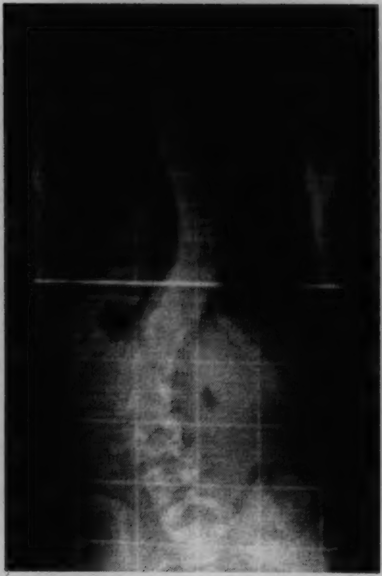


Figure 2-4: Example of a frontal radiograph of a patient with a double scoliotic curve. The Cobb angle of the right thoracic curve is 30°. The Cobb angle of the left lumbar curve is 53°[1].

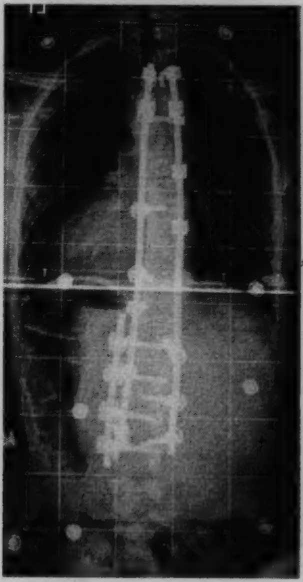


Figure 2-5: Example of a frontal radiograph of the above spine after having undergone successful surgery.[2]

---

[1] Public domain radiograph taken from the Wikipedia.

2.2.3 Magnetic resonance imaging (MRI)

MRI is advisable, especially in children under the age of 12 years, to identify neuroanatomical defects, spinal infections, and tumours [70]. The whole spine should be imaged in a lateral view [7].

2.2.4 Other investigations

Computed Tomography imaging might be used to examine the spinal instrumentation in the unlikely case of post-operative paraplegia [70] or to examine the spinal curve where there are multiple congenital defects [7].

Nuclear scintigraphy is sometimes used to detect pseudarthrose along the fused spine [70].

2.3 Medical image registration

Medical image registration [24], [27], [84], [57] is the determination of a geometrical transformation that aligns anatomical structures of interest in one image with corresponding anatomical structures in another image. Medical image registration has a wide range of potential applications. These include [24]:

---

[2] Public domain radiograph taken from the Wikipedia. Both radiographs, before and after surgery, are available at http://en.wikipedia.org/wiki/Scoliosis . Last accessed on November 04, 2007.

- Monitoring changes, over time, in the anatomical structure of interest. This is achieved by registering images of the same modality (intra-modality) and same patient (intra-subject) acquired at different points in time.

- Combining information, of the same patient, from multiple imaging modalities in a clinically meaningful way. This is achieved by registering images of different modality (inter-modality) and same patient (intra-subject) taken over relatively short periods.

- Relating pre-intervention images to the physical reality of the patient during image-guided interventions. This is achieved by registering images of different modality (inter-modality) and same patient (intra-subject).

- Comparing one individual's anatomy with another or with a standardized atlas. This is achieved by registering images of the same modality (intra-modality) and different patients (inter-subject).

The initial step before applying any registration method involves selecting the type of information that will be used in the matching of the two images [24]. This information is based either on features or on intensity values. The corresponding registration method is termed either feature-based or intensity-based. In feature-based registration, the features represent anatomical structures or artificial objects added to the patient. These objects could be invasive (such as bone implanted markers) or non-invasive (such as skin attached markers). Common features used in the alignment of medical images include pairs of points, edges, gradient magnitudes, contours, surfaces, and principal axes. In feature-based registration, errors in the feature extraction will affect negatively the registration method. Intensity-based registration methods use intensity values or a function of the intensity values for the alignment of the images. Intensity-based registration methods (the maximization of mutual information [44], [79], [68], in particular), have been successfully applied in a number of medical image registration problems because there is no need to extract features from the images. However, because intensity-based registration methods do not, in general, incorporate spatial information, their robustness is questionable [52].

2.3.1 Registration components

The main components of any image registration algorithm are geometrical transformation, similarity measure, optimization strategy, and interpolation method [24], [57].

- Geometrical Transformations. Geometrical transformations align corresponding objects in two or more images [24]. The images could be two dimensional (2-D) or three dimensional (3-D), so the transformation could map points from a 2-D space to a 2-D space, from a 3-D space to a 3-D space, or between a 3-D space and a 2-D space [24]. The transformation can have several forms: rigid, similarity, affine, projective, or curved (Figure 2-6 shows an example in 2-D). Rigid transformations consist of rotation and translation only. Similarity transformations consist of rotation, translations, and uniform scaling. Affine transformations consist of rotations, translations, shearing and scaling. Projective transformations map straight lines to straight lines but parallelism is in general not preserved. Curved transformations may map straight lines to curves [27].

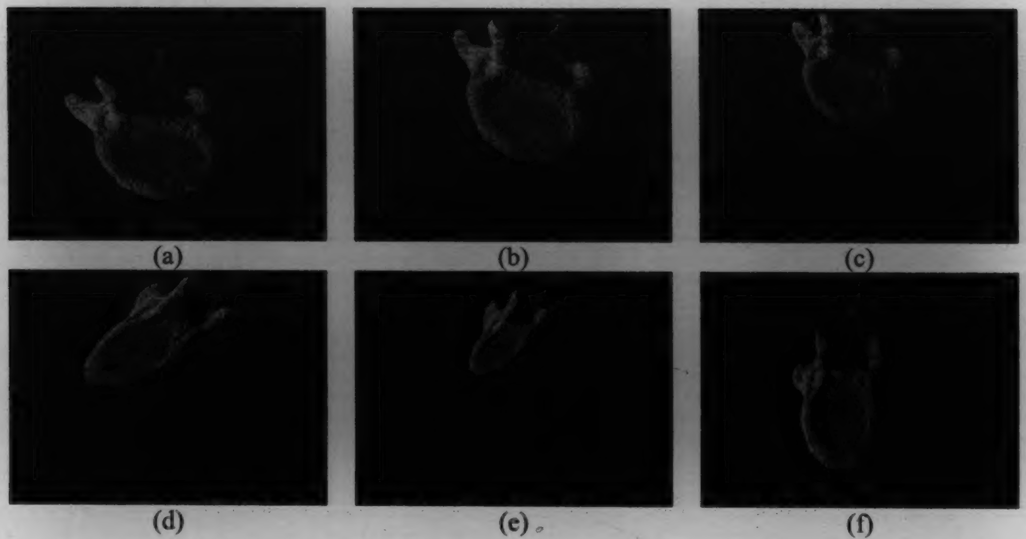


Figure 2-6. Examples of 2D image transformations: (a) original image, (b) image after rigid transformation, (c) image after similarity transformation, (d) image after affine transformation, (e) image after perspective transformation, (f) image after curved transformation

- Similarity Measures. Similarity measures quantify the quality of the match of the two images [27]. There are two types of similarity measures: geometrical similarity measures (used on feature-based registration) and intensity similarity measures (used on intensity-based registration). Geometrical similarity measures involve minimizing cost functions related to the distance between corresponding features in the two images. Intensity similarity measures involve minimizing cost functions computed using the intensity values (directly or indirectly) in regions of interest in the two images [24].

- Interpolation Methods. Interpolation methods estimate pixel intensity values at requested positions resulting from transforming points from one image to another [27]. Three of the most popular interpolator methods are: nearest neighbor, linear, and B-splines. The nearest neighbor interpolation method assigns the intensity value of the spatially closest neighbor. This method has the lowest complexity of implementation but it is the one with the largest loss of quality. The linear interpolation method assigns the intensity value of the weighted average of the surrounding pixels. The weight assigned to each pixel is proportional to its distance to the requested position. The linear interpolation method has a very low complexity of implementation (but higher than that of the nearest neighbor method) without a large loss of quality. B-splines are piece-wise polynomials in which the pieces are smoothly connected together. The B-splines interpolation method assigns the intensity value of the product of the B-splines coefficients with shifted B-splines kernels around the requested location [73].

- Optimization Procedure. Optimization refers to the iterative approach of adjusting the transformation parameters (in the intensity-based registration) or the alignment between features (in feature-based registration) in an attempt to improve (maximize or minimize) the similarity measure [24]. In the feature-based registration, the transformation is computed directly from the correspondences between features. The optimization procedure starts with an initial estimate of the transform (or correspondence). Based on this estimate, the similarity measure is computed. The optimization procedure then makes a new estimate of the transformation parameters, computes the similarity measure, and continues the process until there is not significant improvement in the value of the similarity measure. The estimation of the transformation parameters is done following approaches that use information that is either local or global. Approaches using local information can either use similarity measures' derivatives or not. Examples of optimization approaches using derivatives are gradient descent and quasi-Newton [53]. Examples of optimization approaches that do not use derivatives are downhill search and Powel's method [53]. Approaches using global information include search techniques based on the principles of natural selection, evolution theory, and probability. The main global optimization techniques used for medical image registration are genetic algorithms, simulated annealing, and deterministic annealing [55]. For local optimization procedures to converge to the correct answer, the initial estimate has to be sufficiently close to the expected solution, i.e., the initial estimate has to be within a portion of the parameter space known as the capture range. Because the capture range cannot be known a priori (it depends on the similarity measure and on image properties such as modality, field of view, and contents), it is important to visually inspect the initial estimate to make sure it is close to the correct solution. A further implication of the existence of a capture range is that global optimization methods must be used with caution because they can move outside the capture range.

2.3.2 Registration Process

Having described the main components of an image registration approach, one can summarize the algorithm as illustrated in Figure 2-7 [57]. In this figure, an input image is matched to a reference image. The similarity measure quantifies the goodness of the match. The optimization computes the optimal transformation parameters as a function of the similarity measure. The transformation finds the mapping that relates points in the input image to points in the reference image. Finally, the interpolation evaluates the image intensity values at the mapped positions.

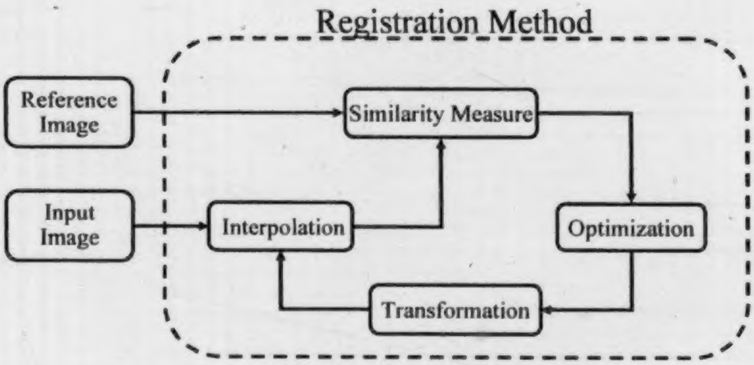


Figure 2-7: Functional blocks of a registration process

2.4 Image registration of spine images

Image registration has been used to perform quantitative analysis of spinal images [25], [26], [32], [60], [75]. Quantitative analysis of spinal images involves the registration of multiple images of the same subject that are taken with the same image modality at different times for monitoring changes or with multiple image modalities for diagnostic purposes. Rogers et al. [60] and Haughton et al. [26] used an intensity-based registration method [24] for rigidly aligning magnetic resonance (MR) images to aid in the diagnosis of spinal instability. Hamadeh and Cinquin [25] used surface-based registration for rigidly registering computed tomography (CT) scans and radiographs to perform kinematic studies of the lumbar spine. Huesman et al. [32] used a combination of paired-point and intensity-based registration methods [44], [79] for aligning CT scans and MR images to study the progression of degenerative diseases in the spine, in particular spinal stenosis. Van Cleynenbreugel et al. [75] rigidly registered preoperative and postoperative CT scans, using an intensity-based registration method, for assessing the accuracy of pedicle screw insertion. The strongest point of the aforementioned studies on quantitative analysis of spinal images is the fact that they provide position and orientation information of anatomical structures of interest. This information is of particular interest in the development of new approaches to diagnose, plan surgery, guide intra-operative procedures, and evaluate treatment outcome. The main limitation of these works is that they do not consider the automated evaluation of registration solutions.

2.5 Assessment of quality of image registration results

The approaches for estimating the goodness of registration solutions can be divided in two categories: those in which a "gold standard" (the correct registration transformation) is known and those in which the "gold standard is not available (as is usually the case in clinical practice). If a "gold standard" is available, the quality of the registration solution can be studied by analyzing the results obtained for one or more initial estimates [71], [51]. If the "gold standard" is not available, the registration performance can be estimated by a domain expert using visual inspection [20], [81]. In [20], the authors performed a study on the efficacy of visual assessment of registration accuracy on CT to MR images. Their results suggest that at least 80% of the time experts can detect misalignments as small as 2 mm, at least 81% of the time experts can detect misalignments as small as 3 mm, and at least 87% of the time experts can detect misalignments as small as 5 mm. In [81], the authors performed a study on the efficacy of visual assessment of registration accuracy on positron emission tomography (PET) to MR images. They considered one axis at the time (only rotation around one axis or translation in an axis at the time). Their results suggest that 100% of the time, experts can detect translational misregistrations of 3 mm or more and rotational misregistrations of 4 degrees or more. They also reported that 60% of the time experts can detect misregistrations of more than 2 mm and that 80% of the time experts could detect misregistrations of more than 3 degrees. The problem with the visual inspection approach is that it is time consuming and experts may not always be available. Moreover, for algorithms that produce a fairly uniform range of errors around the required accuracy, there is a risk that the expert will generate too many false negatives (images that are well registered, but are classified as failures) or too many false positives (images that are not well registered, but are classified as successes) [27]. The registration performance can also be estimated by computing statistical measurement errors based on the location of anatomical landmarks or implanted markers that are easily identifiable [19]. The problem with this approach is that easily identifiable landmarks or markers are not always available as is the case of the spine radiographs used in scoliosis research [11]. Another way of estimating the registration performance involves studying the consistency of the transformation (how close to the Identity transformation the registration of an image triple, A to B, B to C, and C to A) [30]. The problem with the consistency measurement is that it underestimates the true error of the registration algorithm [27]. Recently, approaches based on image segmentation have been proposed to evaluate the quality of the registration results [59], [15], [48], [64]. In [59], a three-phase process (segmentation of the images, computation of principal axes, and determination of a quality measure from contour volumes) was used to distinguish correctly registered solutions from greatly misregistered solutions. Correctly registered solutions were defined as those in which the translation error was less than or equal to 3 mm and the rotation error was less than or equal to 4 degrees following the results presented in [81]. The best result they reported had a sensitivity of 1.0 for a specificity of about 0.6. In [64], a process of matching edges in small regions of interest was used to register 2-D single-modality images of outdoor scenes. They included uncertainty analysis to determine whether a registration solution was acceptable or not. In that work, two noise-contaminated versions of each one of three outdoor scene images were created for three levels of SNR (SNR=3, SNR=2, and SNR=1). One of the images in each pair was translated by a random number of pixels in the range [-4, 4] and an attempt to solve the registration problem (the registration of one noisy image with the translated version of the same noisy image) was carried. The simulations were replicated 200 times. The results reported for the registration of noisy images with a SNR=3 indicate that the accuracy was between 86% and 100% depending on the images being analyzed. In [48], the authors presented an approach to assess the quality of the results of a registration algorithm for CT and MR images of the head. The process consisted of five phases: (1) registration of the CT and MR images, (2) segmentation of the CT image, (3) mapping of the labels of the CT image to the MR image resulting in an MR sub-volume labeled S_{MR1}, (4) segmentation of the MR image resulting in an MR sub-volume labeled S_{MR2}. Pixels that belonged to both S_{MR1} and S_{MR2} were classified as safe (high registration accuracy) and colored green. Pixels that belonged to S_{MR1} but not S_{MR2} were classified as unsafe (low registration accuracy) and colored red. (5) The results were visualized in a fused CT–MR image volume, making it easy for the human operator to identify the regions of low and high registration accuracy. The results reported indicate that 99% of the unsafe pixels were classified as unsafe and between 65% and 100% of the safe pixels (depending on the level of misregistration) were classified as safe. In [15], Crum et al., proposed a generalized overlap measures for evaluation of registration and segmentation results. The measures were based on the computation of the overlap of segmentation labels. For evaluating image registration solutions, the authors suggested comparing the segmentations of the images before and after registration to assess how well the registration brought them into alignment. However, the authors did not report any results on evaluating registration quality. The results presented were on image segmentation evaluation. The main problem with the approaches in [59], [64], [48], [15] is that they require the segmentations to be of high quality to avoid the propagation of segmentation errors into the registration evaluation.

2.6 Summary

This chapter has provided an overview of the theoretical background associated with this work. This chapter has given a brief description of fuzzy sets, a list of operations available for fuzzy sets, and the relation between fuzzy sets and digital images. The main types of imaging modalities used for studying spinal deformities were introduced. Medical image registration was presented as well as the components of an image registration application. The common steps of the image registration process were also presented. A review of image registration for the quantitative analysis of spine images was also presented. Finally, the approaches for assessing the quality of a registration solution were described.

The next chapter will present the proposed fuzzy computing approach for controlling image registration applications.

3 Combining intensity-based and gradient-based fuzzy Jaccard indices for the design of an image registration similarity measure[3]

In this chapter, with the goal of incorporating spatial information in an intensity-based image registration algorithm, a novel similarity measure is introduced and the proof that it is indeed a similarity measure is presented. The similarity measure estimates the goodness of the match of two images by computing the overlap of a fuzzy set [82] representation of gradient and intensity values.

3.1 Digital images and fuzzy sets

A fuzzy set A in a universe of discourse $\mathbf{X} = \{ x_1, x_2, \ldots, x_n \}$ is defined by a mapping from the universe of discourse to the interval [0,1]. Thus, a fuzzy set A in **X** may be represented as a set of ordered pairs $\{ (x_1, A(x_1)), (x_2, A(x_2)), \ldots, (x_n, A(x_n)) \}$, where $A(x_i)$ describes a degree of membership of x_i in A. Digital images can be identified with fuzzy sets that take values on the grid points (i, j). In this work, fuzzy sets are used to represent the brightness level in the images. The brightness levels might vary from a state of no brightness (with a membership degree of zero) to a state of complete brightness (with a membership degree of one). Brightness levels in between these two extremes would get increasingly lighter as they move from black to white (and would have membership values between 0 and 1). Using the notation of fuzzy sets, one can write the fuzzy set representation of a digital image as an M x N array,

---

[3] A version of this chapter has been submitted for publication. L. Ramirez, N. Durdle, J. Raso, Medical & Biological Engineering & Computing. 2007.

$$A = \begin{bmatrix} A(x_{11}) & A(x_{12}) & \cdots & A(x_{1j}) & \cdots & A(x_{1N}) \\ A(x_{21}) & A(x_{22}) & \cdots & A(x_{2j}) & \cdots & A(x_{2N}) \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ A(x_{i1}) & A(x_{i2}) & \cdots & A(x_{ij}) & \cdots & A(x_{iN}) \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ A(x_{M1}) & A(x_{M2}) & \cdots & A(x_{Mj}) & \cdots & A(x_{MN}) \end{bmatrix} \tag{3-1}$$

where $i, j \in \aleph$, $0 \le i \le M$ and $0 \le j \le N$ with M and N being the dimensions of the digital image. $A(x_{ij})$ ($0 \le A(x_{ij}) \le 1$) represents the membership value of the (i,j)th pixel to the fuzzy set "brightness level". By representing a digital image as a fuzzy set, one can use all the operations on fuzzy sets to deal with the imprecision, uncertainty, and ambiguity characteristics of images.

3.2 Fuzzy sets similarity measures for medical image registration

Medical image registration [24], [27] is the determination of a geometrical transformation that aligns anatomical structures of interest in one image with corresponding anatomical structures in another image. In image registration, similarity measures quantify the quality of the match of the two images. Because in this thesis fuzzy sets are used to represent digital images, one can use any of the similarity measures defined in Section 2.1.5 to control the image registration process. In this thesis, similarity between fuzzy sets is defined in this way: Let $F(x)$ be the class of all fuzzy sets of **X** and let S be a mapping $S : F(x) \times F(x) \rightarrow [0,1]$

Definition: $S(A,B)$ is said to be a degree of similarity between $A \in F(x)$ and $B \in F(x)$ if $S(A,B)$ satisfies the following properties [49], [18], [9], [10]:

- P1 (Boundary Conditions): $0 \le S(A,B) \le 1$
- P2: (Reflexivity): if $A = B,\ S(A,B) = 1$
- P3 (Commutativity): $S(A,B) = S(B,A)$
- P4 (Transitivity): if $A \subseteq B \subseteq C \quad A,B,C \in F(x)$ then
  $S(A,C) \le S(A,B)$ and $S(A,C) \le S(B,C)$

The proposed fuzzy similarity measure (*SM*) to be used for image registration is [49]

$$SM(A,B) = \frac{card(A \cap B)}{card(A \cup B)} \tag{3-2}$$

with *card* being the cardinality of a fuzzy set (i.e., the sum of membership functions of the elements in the fuzzy set equation (3-3))and $A \cap B$ and $A \cup B$ being the fuzzy intersection equation (3-4) and the fuzzy union equation (3-5) respectively. The entire expression, therefore, represents the ratio of the sum of the membership values of the elements in the fuzzy intersection of *A* and *B* to the sum of the membership values of the elements in the fuzzy union of *A* and *B*. It is important to note that *SM* is dimensionless.

$$card(A) = \sum_{x \in \mathbf{X}} A(x) \tag{3-3}$$

$$A \cap B : (A(x) \cap B(x)) = A(x)\,\mathrm{t}\,B(y) \tag{3-4}$$

$$A \cup B : (A(x) \cup B(x)) = A(x)\,\mathrm{s}\,B(y) \tag{3-5}$$

SM is a fuzzy sets generalization (by taking into account partial memberships of the elements in the sets) of the Jaccard Index or Tanimoto Coefficient [15], referred in this thesis as the Fuzzy Jaccard Index (*FJI*). The *FJI* produces a smooth transition from equal ($FJI(A,B) = 1$) to completely non-equal fuzzy sets ($FJI(A,B) = 0$). It is based on the fuzzy set-theoretic operations of intersection and union, which are defined for a given pair of **t**-norm and **s**-norm. In this thesis, *min* will be used for **t**-norm and *max* for **s**-norm. These norms were chosen because of their low computational cost. Low computational cost in a similarity measure is necessary for image registration because the similarity measure might be computed several hundreds (or even several thousands) times during the optimization process. The *FJI* can be written as

$$FJI(A,B) = \frac{\sum_{x \in \mathbf{X}} min(A(x), B(x))}{\sum_{x \in \mathbf{X}} max(A(x), B(x))} \qquad (3\text{-}6)$$

The choice of *FJI* as the basis for the proposed image registration similarity measure is based on the following observations:

- First, image registration is aimed at finding the transformation that would increase the overlap between structures of interest in the images under study. *FJI* is based on one of the most frequently used measures for the evaluation of the overlap in segmentation results, the Jaccard Index [15], [74]. Therefore, *FJI* is a good candidate for controlling the image registration process.

- Second, a step in the image registration process (Section 2.3.2) involves the comparison, with a similarity measure, of the reference image with a transformed version of the input image. Recently, the *FJI* has been successfully used in the comparison of images for the evaluation of algorithms designed to solve problems, such as noise reduction, deblurring, and compression [16].

- Third, Crum et al. [15] proposed a similarity measure based on the *FJI* for the registration of segmented (labelled) images. This suggests that in fact the *FJI* is a valid alternative as a similarity measure for medical image registration. It is important to note that in the approach by Crum et al. [15], the success of the registration is highly dependent on the quality of the segmentation and therefore does not always produce satisfactory results. To cope with this drawback, a similarity measure that does not require the segmentation of the images is proposed.

3.2.1 Fuzzy Jaccard Index on intensity values

Given two overlapping images *A* and *C*=*M*(*B*), with *M*(*B*) being the result of an unknown misalignment *M* to the image *B*, the Fuzzy Jaccard Index on intensity values (FJI_i) is computed by

$$FJI_i(A,C) = \frac{\sum_{x\in X} min(A_i(x),C_i(x))}{\sum_{x\in X} max(A_i(x),C_i(x))} \qquad (3\text{-}7)$$

with $A_i(x)$ and $C_i(x)$ being fuzzy set representations of the intensity values of *A* and *C* respectively. The fuzzy set representation of intensity values (brightness) for *A* is computed by a mapping function defined by the following equation:

$$A_i(x) = \frac{A(x) - min(A)}{max(A) - min(A)} \qquad (3\text{-}8)$$

Similarly, the fuzzy set representation of intensity values (brightness) for C is computed by a mapping function defined by the following equation:

$$C_i(x) = \frac{C(x) - min(C)}{max(C) - min(C)} \tag{3-9}$$

The membership values of $A_i(x)$ and $C_i(x)$ denote the degree of brightness relative to the intensity values of A and C. FJI_i is a fuzzy similarity measure that quantifies the fuzzy overlap between fuzzy set representations of the brightness in both images. The aim of FJI_i is to exploit the functional relation between sets of pixels having similar intensity values in both images. Registration is assumed to correspond to maximizing the FJI_i. The images have to be aligned in such a manner that the amount of overlap between similar intensity values is maximized. The assumption that the images being registered are similar, except for the misalignment, is valid for image of the same modality that

- differ only by Gaussian noise,
- are used in serial studies (in which it is expected that the images being aligned will be identical except for small changes resulting from disease progression or response to treatment), or
- are taken during the same study to correct for patient movement.

FJI_i might fail if the data diverges too much from the cases presented above. For example, if a small number of pixels change intensity by a large amount, they can have a large effect on the change in FJI_i. See, for instance, the example presented in Table 3-1. In the example, the similarity measure is computed for a Reference Image and three input images. In the first case, the Input Image is identical to the Reference Image. The resulting FJI_i is, as expected, equal to 1.0 (which is the maximum possible value). In the second case, the Input Image is a shifted version of the Reference Image. The resulting FJI_i is equal to 0.43. The Input Image, in the third case, has the same shift as the Input Image in the second case. In the third case, however, three pixels also changed intensity by a large amount. As a result, the FJI_i is smaller than the one in the second case. The new FJI_i is equal to 0.24 (equivalent to a reduction of 44% in the value of FJI_i).

	Intensity values of sample images						Fuzzy sets representations						FJI_i(Ref., Input)
Reference	0	5	50	50	70	0	0.00	0.02	0.20	0.20	0.27	0.00	
Input 1	0	5	50	50	70	0	0.00	0.02	0.20	0.20	0.27	0.00	1.00
Input 2	0	0	5	50	50	70	0.00	0.00	0.02	0.20	0.20	0.27	0.43
Input 3	0	0	5	150	150	170	0.00	0.00	0.02	0.59	0.59	0.67	0.24

Table 3-1: Example of FJI_i computation. The column labelled Intensity values of sample images contains the images to be registered. The column labelled Fuzzy sets representations contains membership values denoting the degree of brightness of the pixels in the Sample Images. The column labelled FJI_i contains the Fuzzy Jaccard Indices for the Reference and Input Images.

To reduce the effect of changes in intensity value, a new similarity measure that involves the computation of the Fuzzy Jaccard Index on gradient values instead of intensity values is presented.

3.2.2 Fuzzy Jaccard Index on gradient values

Given two overlapping images A and $C=M(B)$, with $M(B)$ being the result of an unknown misalignment M to the image B, the Fuzzy Jaccard Index on gradient magnitude values (FJI_g) is computed by

$$FJI_g(A,C)=\frac{\sum_{x\in \mathbf{X}} min\left(A_g(x),C_g(x)\right)}{\sum_{x\in \mathbf{X}} max\left(A_g(x),C_g(x)\right)} \qquad (3\text{-}10)$$

with $A_g(x)$ and $C_g(x)$ being fuzzy set representations of the gradient magnitude values of A and C respectively. The gradient is a vector that is perpendicular to the edges in the image and that contains the partial derivatives in all directions [34]. The magnitude of the gradient is a measure of the strength of the edges independently of its direction [34]. Because digital images are a collection of discrete values on a discrete grid and are not based on an analytical function, there is no set of rules on how to compute the derivatives. This is why the partial derivatives are usually approximated by discrete differences [34] at a certain scale in which the differentiation should be performed. This is usually done by preprocessing the image with a smoothing filter and then applying a differential operator. In this work, the magnitude computed on the discrete differences will be called gradient magnitude following the convention adopted in the image registration community [33]. The most popular filter for doing the image smoothing is the Gaussian kernel. By choosing a particular value for the standard deviation (σ) of the Gaussian, an associated scale is selected that removes high image frequencies content and has a smoothing effect on the image [33]. Figure 3-1 shows some examples of scaled gradient magnitude images. Note that more and more details vanish from the images as the value of σ increases.

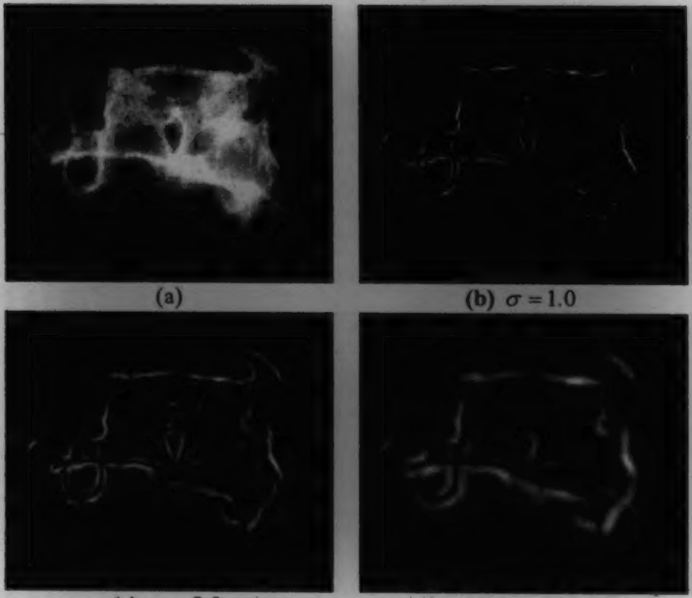
The process of computing a fuzzy sets representation of gradient magnitudes is as follows. First, the gradients for the input and reference images are computed at a certain scale (σ). Then, a fuzzy set representation of gradient magnitude values is computed by linearly scaling the gradients between 0 and 1 with the following equation:

$$A_g(x) = \frac{|\nabla_\sigma A(x)| - min(|\nabla_\sigma A|)}{max(|\nabla_\sigma A|) - min(|\nabla_\sigma A|)} \qquad (3\text{-}11)$$

with $|\nabla_\sigma A|$ being the gradient magnitude of the image A at a given scale σ. Similarly, the fuzzy set representation of gradient magnitude values (edgeness) for C is computed by a mapping function defined by the following equation:

$$C_g(x) = \frac{|\nabla_\sigma C(x)| - min(|\nabla_\sigma C|)}{max(|\nabla_\sigma C|) - min(|\nabla_\sigma C|)} \tag{3-12}$$

The membership values of $A_g(x)$ and $C_g(x)$ denote the degree of edgeness relative to the gradient magnitude values of A and C. FJI_g is a fuzzy similarity measure that quantifies the fuzzy overlap between fuzzy set representations of the gradient magnitude values in both images. The aim of FJI_g is to exploit the functional relation between sets of pixels having large gradient magnitude in both images. Registration is assumed to correspond to maximizing the FJI_g. The images have to be aligned in such a manner that the amount of overlap between similar gradient magnitude values is maximized. The assumption that the images being registered have similar gradient magnitude values is valid not only for images of the same modality but also for images of different modalities that depict the same tissue transitions.



(a) (b) $\sigma = 1.0$

(c) $\sigma = 2.0$ (d) $\sigma = 5.0$

Figure 3-1: Gradient magnitude images of a lumbar vertebra image. (a) Original image. (b), (c), and (d): gradient magnitude images at scales $\sigma = 1.0$, $\sigma = 2.0$, and $\sigma = 5.0$ respectively.

FJI_g can fail if the images do not depict the same tissue transitions which may cause strong gradients in one image modality to be absent or less prominent in the other modality. Unlike FJI_i, if a small number of pixels change intensity by a large amount, they would have a moderate effect on the change in FJI_g. See for instance the example presented in Table 3-2. In the example, the similarity measure is computed for the same Reference Image and input images shown in Table 3-1. The Gradient Images are computed for σ=1.0. In the first case, the Input Image is identical to the Reference Image. The resulting FJI_g is, as expected, equal to 1.0 (which is the maximum possible value). In the second case, the Input Image is a shifted version of the Reference Image. The resulting FJI_g is equal to 0.46. The Input Image, in the third case and the Input Image in the second case have the same shift. In the third case, however, three pixels also changed intensity by a large amount. As a result, the FJI_g is smaller than the one in the second case. The new FJI_g is equal to 0.35 (equivalent to a reduction of 23% in the value of FJI_g).

Comparing the effect of large changes in the intensity values of a small number of pixels when computing FJI_i and FJI_g, it looks like the FJI_g might be more robust. The reduction in the value of FJI_g was 23% compared to a reduction of 44% in the value of FJI_i for the same displacement and same change in intensity value.

	Gradient magnitudes of sample images						Fuzzy sets representations						FJI_g(Ref., Input)
Reference	41	192	193	0	180	255	0.16	0.75	0.76	0.00	0.71	1.00	
Input 1	41	192	192	0	180	255	0.16	0.75	0.76	0.00	0.71	1.00	1.00
Input 2	0	85	241	255	162	93	0.00	0.33	0.95	1.00	0.64	0.36	0.46
Input 3	0	81	250	255	107	31	0.00	0.32	0.98	1.00	0.42	0.12	0.35

Table 3-2: Example of FJI_g computation. The column labelled Gradient magnitudes of sample images contains the gradient magnitudes of the images to be registered. The column labelled Fuzzy sets representations contains membership values denoting the degree of edgeness of the pixels in the Sample images. The column labelled FJI_g contains the Fuzzy Jaccard Indices for the Reference and Input Images. The Reference and Input images used in this example were obtained by finding the gradient magnitude of the images in Table 3-1.

3.2.3 The *FJI* as a fuzzy sets similarity measure

Before using the proposed similarity measure in image registration applications, it is important to prove that the Fuzzy Jaccard Index is indeed a fuzzy sets similarity measure as defined at the beginning of Section 3.2. The following proof follows an approach similar to that suggested in [18].

Proof: Given two fuzzy sets A and B defined in the same universe of discourse **X**, to be a similarity measure, any functional has to satisfy the properties described at the beginning of Section 3.2. That is

- P1 (Boundary Conditions): $0 \leq S(A,B) \leq 1$
- P2: (Reflexivity): if $A = B,\ S(A,B) = 1$

- P3 (Commutativity): $S(A,B)=S(B,A)$
- P4 (Transitivity): if $A \subseteq B \subseteq C \quad A,B,C \in F(x)$ then

  $S(A,C) \le S(A,B)$ and $S(A,C) \le S(B,C)$

P1: First, recall the definition of $FJI(A,B)=\frac{\sum_{x\in \mathbf{X}} min(A(x),B(x))}{\sum_{x\in \mathbf{X}} max(A(x),B(x))}$, then because $0 \le min(A(x),B(x)) \le max(A(x),B(x))$, one has that $0 \le FJI(A,B) \le 1$

P2: First, recall the definition of $FJI(A,B)=\frac{\sum_{x\in \mathbf{X}} min(A(x),B(x))}{\sum_{x\in \mathbf{X}} max(A(x),B(x))}$, then if $A=B$

$min(A(x),B(x)) = min(A(x),A(x)) = A(x)$ and $max(A(x),B(x)) = max(A(x),A(x)) = A(x)$, one has that

$$FJI(A,B)=FJI(A,A)=\frac{\sum_{x\in \mathbf{X}} min(A(x),A(x))}{\sum_{x\in \mathbf{X}} max(A(x),A(x))}=\frac{\sum_{x\in \mathbf{X}} A(x)}{\sum_{x\in \mathbf{X}} A(x)}=1$$

$\therefore FJI(A,B)=1$

P3: First, recall the definition of $FJI(A,B)=\frac{\sum_{x\in \mathbf{X}} min(A(x),B(x))}{\sum_{x\in \mathbf{X}} max(A(x),B(x))}$ and

$FJI(B,A)=\frac{\sum_{x\in \mathbf{X}} min(B(x),A(x))}{\sum_{x\in \mathbf{X}} max(B(x),A(x))}$, then because $min(A(x),B(x)) = min(B(x),A(x))$ and $max(A(x),B(x)) = max(B(x),A(x))$, one has that

$$FJI(A,B)=\frac{\sum_{x\in \mathbf{X}} min(A(x),B(x))}{\sum_{x\in \mathbf{X}} max(A(x),B(x))}=\frac{\sum_{x\in \mathbf{X}} min(B(x),A(x))}{\sum_{x\in \mathbf{X}} max(B(x),A(x))}=FJI(B,A)$$

$\therefore FJI(A,B)=FJI(B,A)$

P4: First, recall the definition of $FJI(A,B)$, $FJI(A,C)$, and $FJI(B,C)$

$$FJI(A,B)=\frac{\sum_{x\in\mathbf{X}} min(A(x),B(x))}{\sum_{x\in\mathbf{X}} max(A(x),B(x))} \tag{3-13}$$

$$FJI(A,C)=\frac{\sum_{x\in\mathbf{X}} min(A(x),C(x))}{\sum_{x\in\mathbf{X}} max(A(x),C(x))} \tag{3-14}$$

$$FJI(B,C)=\frac{\sum_{x\in\mathbf{X}} min(B(x),C(x))}{\sum_{x\in\mathbf{X}} max(B(x),C(x))} \tag{3-15}$$

Then recall that if $A \subseteq B$ then

$$A(x) \le B(x), \forall x \in \mathbf{X} \tag{3-16}$$

if $B \subseteq C$ then

$$B(x) \le C(x), \forall x \in \mathbf{X} \tag{3-17}$$

and if $A \subseteq C$ then

$$A(x) \le C(x), \forall x \in \mathbf{X} \tag{3-18}$$

Using (3-16):

$$min(A(x),B(x)) = A(x) \tag{3-19}$$

and

$$max(A(x), B(x)) = B(x) \tag{3-20}$$

Using (3-17):

$$min(B(x), C(x)) = B(x) \tag{3-21}$$

and

$$max(B(x), C(x)) = C(x) \tag{3-22}$$

Using (3-18):

$$min(A(x), C(x)) = A(x) \tag{3-23}$$

and

$$max(A(x), C(x)) = C(x) \tag{3-24}$$

Using (3-19) and (3-20) in (3-13):

$$FJI(A, B) = \frac{\sum_{x \in \mathbf{X}} min(A(x), B(x))}{\sum_{x \in \mathbf{X}} max(A(x), B(x))} = \frac{\sum_{x \in \mathbf{X}} A(x)}{\sum_{x \in \mathbf{X}} B(x)} \tag{3-25}$$

Using (3-23) and (3-24) in (3-14):

$$FJI(A, C) = \frac{\sum_{x \in \mathbf{X}} min(A(x), C(x))}{\sum_{x \in \mathbf{X}} max(A(x), C(x))} = \frac{\sum_{x \in \mathbf{X}} A(x)}{\sum_{x \in \mathbf{X}} C(x)} \tag{3-26}$$

From (3-17) in (3-25) and (3-26):

$$FJI(A,C)=\frac{\sum_{x\in X} A(x)}{\sum_{x\in X} C(x)}\leq\frac{\sum_{x\in X} A(x)}{\sum_{x\in X} B(x)}=FJI(A,B) \tag{3-27}$$

$$\therefore FJI(A,C)\leq FJI(A,B)$$

Using (3-21) and (3-22) in(3-15):

$$FJI(B,C)=\frac{\sum_{x\in X} min(B(x),C(x))}{\sum_{x\in X} max(B(x),C(x))}=\frac{\sum_{x\in X} B(x)}{\sum_{x\in X} C(x)} \tag{3-28}$$

From (3-16) in (3-26) and (3-28):

$$FJI(A,C)=\frac{\sum_{x\in X} A(x)}{\sum_{x\in X} C(x)}\leq\frac{\sum_{x\in X} B(x)}{\sum_{x\in X} C(x)}=FJI(B,C) \tag{3-29}$$

$$\therefore FJI(A,C)\leq FJI(B,C)$$

Q.E.D.

3.3 The Combined Fuzzy Jaccard Index (*CFJI*)

The *CFJI* combines the proposed overlap measure on gradient magnitudes (the *FJI* on gradients or FJI_g) with a version of the proposed overlap measure that computed the overlap on intensity values (the *FJI* on intensities or FJI_I). As a result, the new similarity measure focuses on simultaneously aligning corresponding edges and corresponding regions of smooth intensity values.

Crum et al. [15] proposed a similarity measure based on the *FJI* for the registration of segmented (labelled) images. In their case, the success of the registration is highly dependent on the quality of the segmentation and therefore does not always produce satisfactory results. To cope with this drawback, a feature extraction methodology that allows the use of the *FJI* while effectively exploiting the available information in the images is proposed in this work. The main feature set considered is the edges of the images because they contain useful structural information about the boundaries of the objects of interest. In particular, the focus is on the gradient magnitude. The gradient magnitude measures the local steepness of the intensity landscape, which has local maxima at the edges. However, working only with gradients would discard a lot of information from the intensity values in the images. For this reason, a second feature set consisting of the intensity values in the images is considered. This second feature set takes into account the functional relation between sets of pixels having similar intensity values in both images. Therefore, the goal of the proposed similarity measure is to align, simultaneously, corresponding edges and corresponding regions of smooth intensity values.

Registration is assumed to correspond to maximizing the Combined Fuzzy Jaccard Index (*CFJI*). The images have to be aligned in such a manner that the amount of overlap between structures of interest is maximized. The *CFJI* and the intersection are computed for the overlapping parts of the images. A problem that can occur when using the intersection on its own is that there is no simple guideline to determine which values are associated with a high degree of alignment. For example, when registering images in which the background is large when compared to the structures of interest, the value of the intersection would be small even for high degrees of alignment. The *CFJI* is better equipped to avoid such problems because it includes the union of the overlapping parts of the images.

The process of computing the proposed similarity measure is as follows. First, the gradients for the input and reference images are computed at a certain scale level (σ). Then, a fuzzy set representation of edges is computed by mapping functions defined by equations (3-11) and (3-12). Next, one can compute the FJI_g by using equation (3-10). After computing FJI_g, it is necessary to compute the portion of the similarity measure that takes into account the functional relation between sets of pixels having similar intensity values in both images. A fuzzy set representation of intensity values is computed by mapping functions defined by equations (3-8) and (3-9). Next, one can compute the FJI_i by using equation (3-7). The final step to compute the proposed similarity measure is to compute a weighted sum (a special case of the models described in [14]) of the FJI_g and FJI_i, i.e.

$$CFJI(A,C) = \alpha \cdot FJI_g(A,C) + (1-\alpha) \cdot FJI_i(A,C),\ 0 \le \alpha \le 1 \tag{3-30}$$

where $\alpha \in [0,1]$ is a weighting factor that controls the impact of FJI_g and FJI_i in the computation of *CFJI*. For $\alpha = 1$, *CFJI* is equal to FJI_g i.e., the fuzzy sets representation of the gradient magnitude values will have the highest impact on the proposed similarity measure. For $\alpha = 0$, *CFJI* is equal to FJI_i i.e., the fuzzy sets representation of the intensity values will have the highest impact on the proposed similarity measure. For values of α in between these two extremes, *CFJI* would get increasingly reliant on the gradient magnitude values as α moves from 0 to 1.

3.3.1 The Combined Fuzzy Jaccard Index (*CFJI*) as a fuzzy sets similarity measure

Given that the FJI was proven to be a fuzzy sets similarity measure, as defined at the beginning of Section 3.2, it is of interest to prove that the *CFJI* is also a fuzzy sets similarity measure. The following proof follows an approach similar to that suggested in [18].

Proof: Given two fuzzy sets A and B defined in the same universe of discourse **X**, to be a similarity measure, any functional has to satisfy the properties described at the beginning of Section 3.2. That is

- P1 (Boundary Conditions): $0 \le S(A,B) \le 1$
- P2: (Reflexivity): if $A = B,\ S(A,B) = 1$
- P3 (Commutativity): $S(A,B) = S(B,A)$
- P4 (Transitivity): if $A \subseteq B \subseteq C \quad A,B,C \in F(x)$ then
  $S(A,C) \le S(A,B)$ and $S(A,C) \le S(B,C)$

P1: First, recall the definition of $CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_i(A,B)$, then because $0 \le \alpha \le 1$, $0 \le FJI_g(A,B) \le 1$, and $0 \le FJI_i(A,B) \le 1$, one has that $0 \le CFJI(A,B) \le 1$

P2: First, recall the definition of $CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_i(A,B)$, then if A=B: $FJI_g(A,A) = 1$ and $FJI_i(A,A) = 1$

$\therefore CFJI(A,A) = \alpha \cdot FJI_g(A,A) + (1-\alpha) \cdot FJI_i(A,A) = \alpha \cdot 1 + (1-\alpha) \cdot 1 = 1$

P3: First, recall the definition of $CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_i(A,B)$ and $CFJI(B,A) = \alpha \cdot FJI_g(B,A) + (1-\alpha) \cdot FJI_i(B,A)$, then because $FJI_g(A,B) = FJI_g(B,A)$ and $FJI_i(A,B) = FJI_i(B,A)$, one has that

$CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_i(A,B) = \alpha \cdot FJI_g(B,A) + (1-\alpha) \cdot FJI_i(B,A) = CFJI(B,A)$

P4: First, based on the definition of *CFJI*:

$CFJI(A,C) = \alpha \cdot FJI_g(A,C) + (1-\alpha) \cdot FJI_t(A,C)$, $CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_t(A,B)$, and $CFJI(B,C) = \alpha \cdot FJI_g(B,C) + (1-\alpha) \cdot FJI_t(B,C)$.

From (3-27):

$$FJI_g(A,C) \le FJI_g(A,B) \tag{3-31}$$

$$FJI_t(A,C) \le FJI_t(A,B) \tag{3-32}$$

From (3-29):

$$FJI_g(A,C) \le FJI_g(B,C) \tag{3-33}$$

$$FJI_t(A,C) \le FJI_t(B,C) \tag{3-34}$$

Using (3-31) and (3-32):

$$CFJI(A,C) = \alpha \cdot FJI_g(A,C) + (1-\alpha) \cdot FJI_t(A,C) \le \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_t(A,B) = CFJI(A,B)$$

$$\therefore CFJI(A,C) \le CFJI(A,B)$$

Using (3-33) and (3-34):

$$CFJI(A,C) = \alpha \cdot FJI_g(A,C) + (1-\alpha) \cdot FJI_t(A,C) \le \alpha \cdot FJI_g(B,C) + (1-\alpha) \cdot FJI_t(B,C) = CFJI(B,C)$$

$$\therefore CFJI(A,C) \le CFJI(B,C)$$

Q.E.D

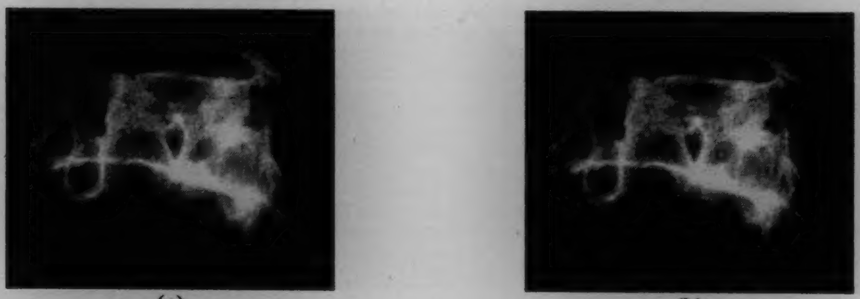
3.3.2 Exploring the sensitivity of the Combined Jaccard Index to various values of α and σ

The *CFJI* measures the amount of area overlap between input and reference images with respect to the total area. However, the value of *CFJI* is affected not only by the amount of misalignment but also by a number of factors such as the type of registration problem being solved (single-modality versus multimodality), the scale factor (σ) used in the computation of gradient, and the weighting factor (α). To illustrate this, two examples exploring the domain of the similarity measure for the single-modality and multimodality registration problems is presented. For the single-modality registration problem, a radiograph of a dry-bone thoracic vertebra is used as input and reference images (Figure 3-2). For the multimodality registration problem, brain images were used, a T1 Magnetic Resonance Image (MRI) as reference image and a Proton Density MRI as input image (Figure 3-9). The algorithms for the implementation were implemented using the C++ programming language and the Insight Toolkit (ITK, www.itk.org) [33]. The corresponding source code is depicted in Appendices 2 and 3.

Single-modality registration problem

These examples aim to illustrate the form of the outputs produced by the proposed similarity measure in a single-modality image registration case. For these simulations, the parametric space to explore consists of a group of translations in a two dimensional (2-D) space. The translations were distributed uniformly in the [-10, 10] pixels interval. The weighting factor α was set to one of three values α=0.0, α=0.5, or α=1.0. The scaling factor σ was set to either σ=1.0 or σ=5.0. The characteristics for α=0.0 and σ=1.0 are depicted in Figure 3-3. The characteristics for α=0.0 and σ=5.0 are depicted in Figure 3-4. The characteristics for α=0.5 and σ=1.0 are depicted in Figure 3-5. The characteristics for α=0.5 and σ=5.0 are depicted in Figure 3-6. The characteristics for α=1.0 and σ=1.0 are depicted in

Figure 3-7. The characteristics for α=1.0 and σ=5.0 are depicted in Figure 3-8. These Figures suggest that for increasing values of α, the baseline value (the value obtained for the largest misregistration applied to the input image) decreases. For increasing values of σ, there is an increase in the number of local maxima. For α=0, the value of σ has no effect on the response of the similarity measure to various levels of misalignment.



(a) **(b)**

Figure 3-2: Dry-bone radiographs of a thoracic vertebra for the single-modality registration problem. (a) Reference image. (b) input image.

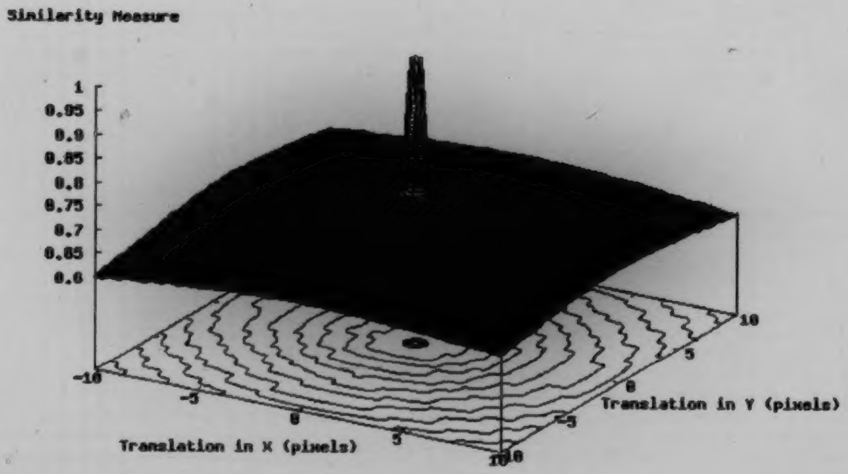


Figure 3-3: Single-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 1.0$.

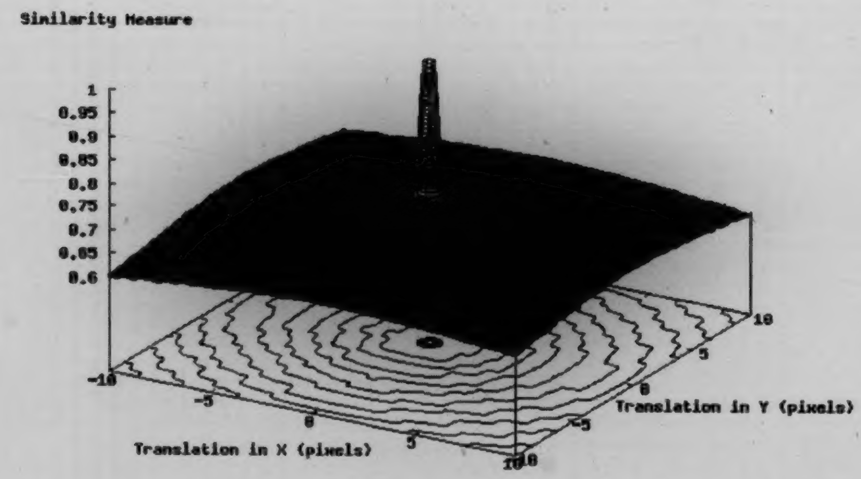


Figure 3-4: Single-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 5.0$

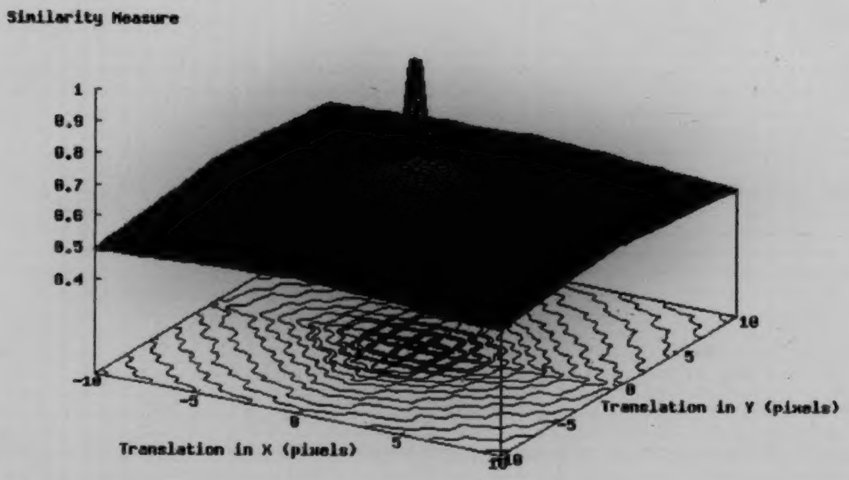


Figure 3-5: Single-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 1.0$.

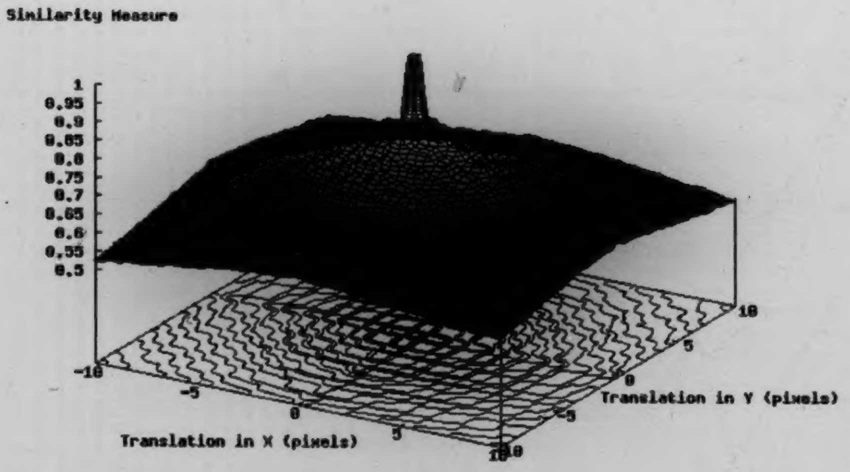


Figure 3-6: Single-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 5.0$.

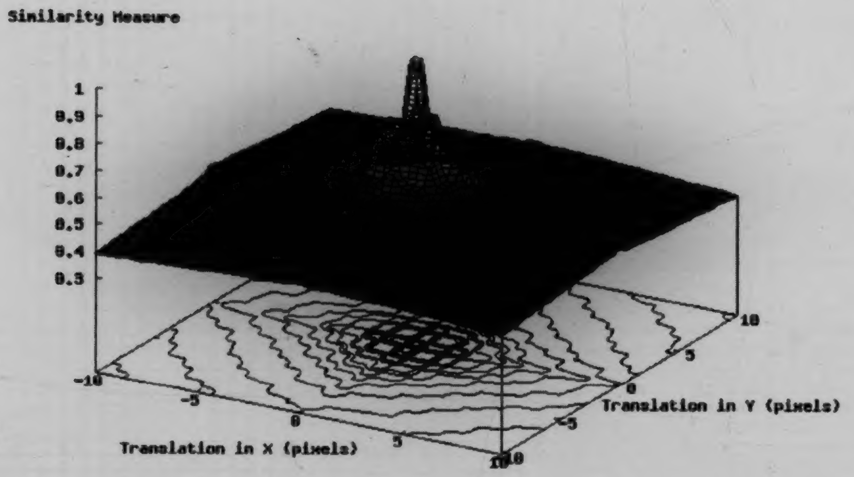


Figure 3-7: Single-modality similarity measure surface plots for $\alpha = 1$ and $\sigma = 1.0$.

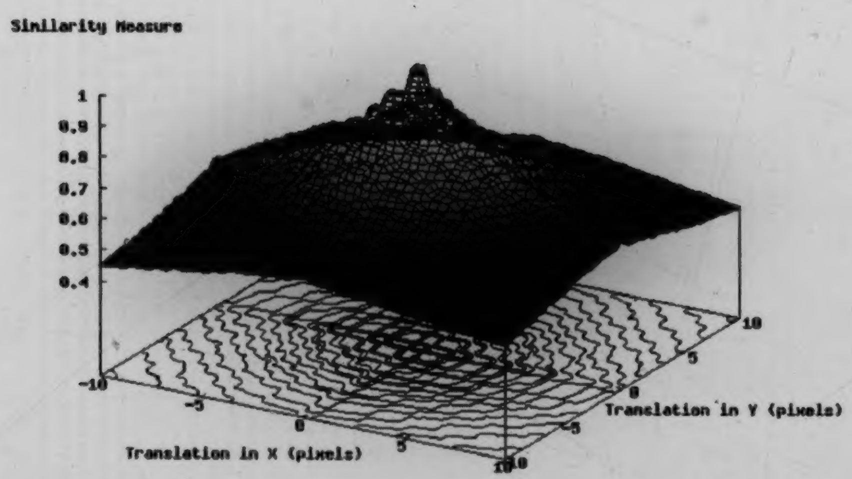


Figure 3-8: Single-modality similarity measure surface plots for $\alpha = 1.0$ and $\sigma = 5.0$.

Multi-modality registration problem

These examples aim to illustrate the form of the outputs produced by the proposed similarity measure in a multi-modality image registration case. For these simulations, the parametric space to explore consists of a group of translations in 2-D that were distributed uniformly in the [-10, 10] pixels interval. The weighting factor α was set to one of three values α=0.0, α=0.5, or α=1.0. The scaling factor σ was set to either σ=1.0 or σ=5.0. The characteristics for α=0.0 and σ=1.0 are depicted in Figure 3-10. The characteristics for α=0.0 and σ=5.0 are depicted in Figure 3-11. The characteristics for α=0.5 and σ=1.0 are depicted in Figure 3-12. The characteristics for α=0.5 and σ=5.0 are depicted in Figure 3-13. The characteristics for α=1.0 and σ=1.0 are depicted in Figure 3-14. The characteristics for α=1.0 and σ=5.0 are depicted in Figure 3-15. These Figures suggest that for increasing values of α, the baseline value (the value obtained for the largest misregistration applied to the input image) decreases. For increasing values of σ, there is an increase in the number of local maxima. For $\alpha=0$, the value of σ has no effect on the response of the similarity measure to various levels of misalignment.

(a)
(b)

Figure 3-9: Brain Magnetic Resonance Images (MRI) for the multi-modality registration problem. (a) Reference image (T1-MRI). (b) Input image (Proton Density MRI).

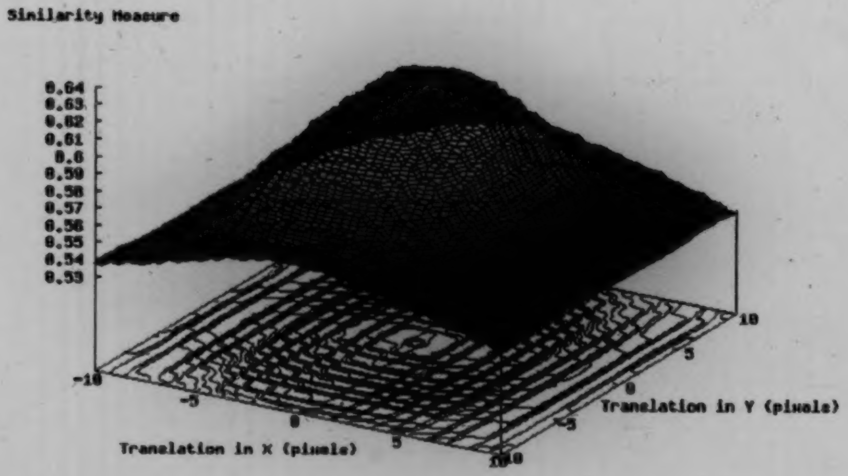


Figure 3-10: Multi-modality similarity measure surface plots for $\alpha=0$ and $\sigma=1.0$.

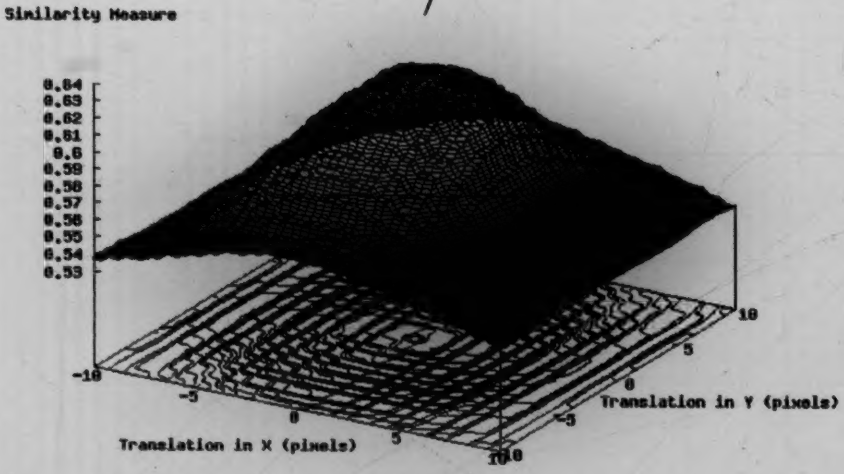


Figure 3-11: Multi-modality similarity measure surface plots for $\alpha = 0$ and $\sigma = 5.0$.

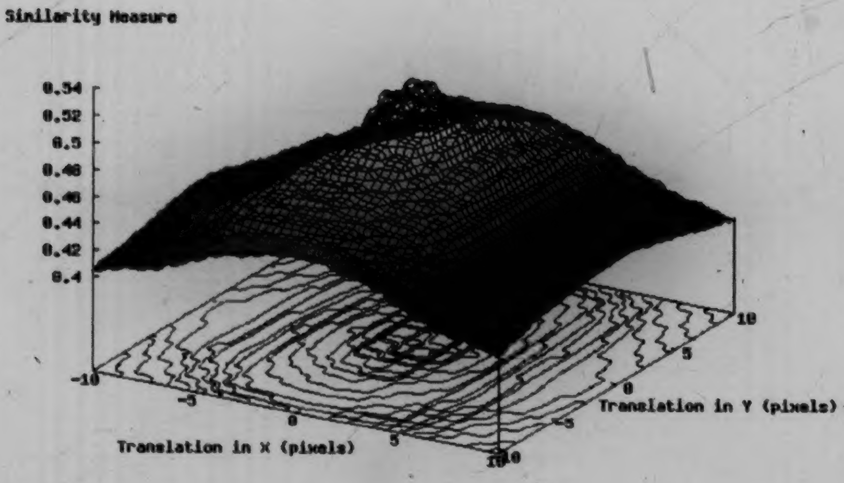


Figure 3-12: Multi-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 1.0$.

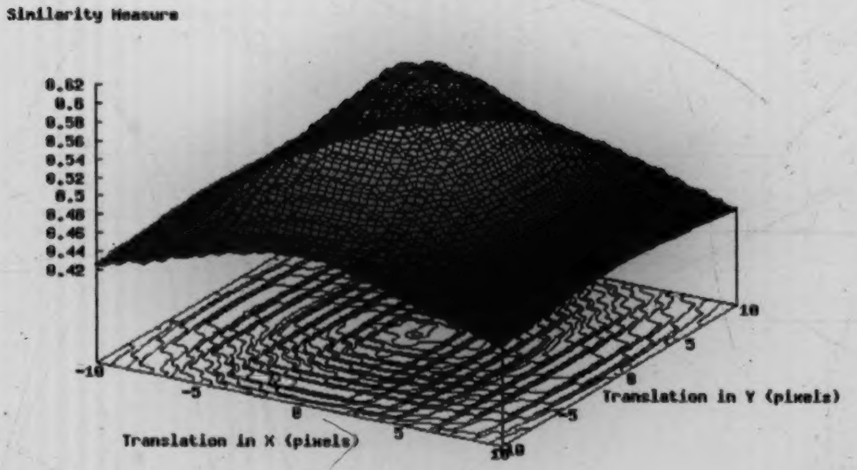


Figure 3-13: Multi-modality similarity measure surface plots for $\alpha = 0.5$ and $\sigma = 5.0$.

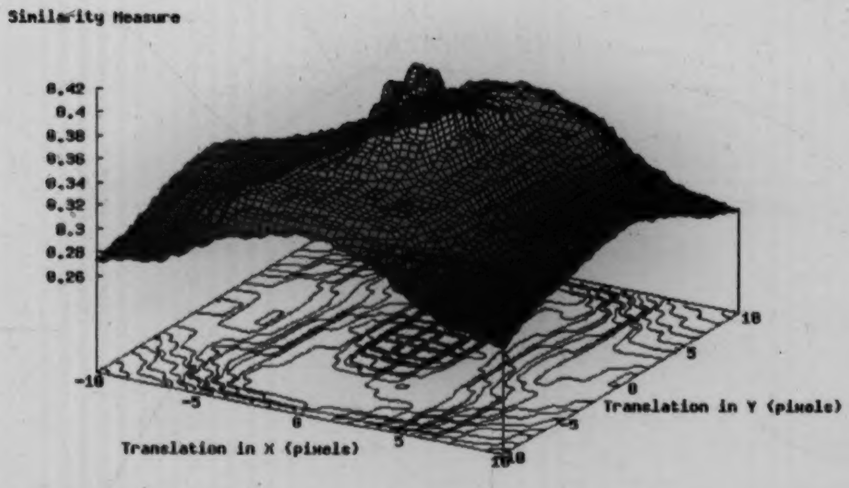


Figure 3-14: Multi-modality similarity measure surface plots for $\alpha = 1$ and $\sigma = 1.0$.

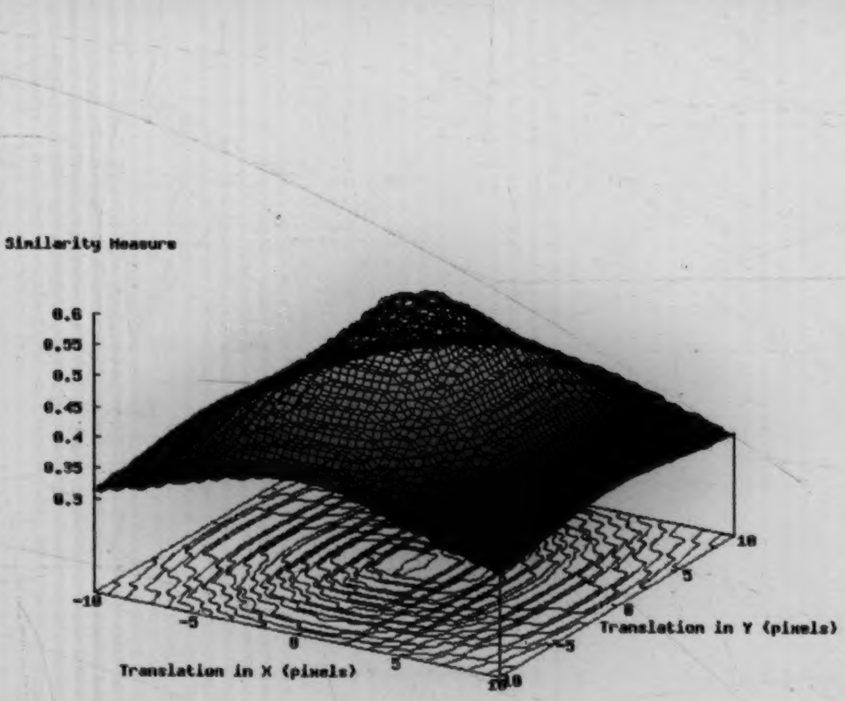


Figure 3-15: Multi-modality similarity measure surface plots for $\alpha = 1.0$ and $\sigma = 5.0$.

3.4 Study of the performance of the *CFJI* in single-modality, multimodality, and model-to-image registration problems

The aim of these simulations was to study the performance of the proposed similarity measure not only on model-to-image registration (which is the focus of this thesis) but also on single-modality and multimodality registration problems. The goal was to verify the validity of the hypothesis that registration algorithms based on the *CFJI* can perform well in a variety of registration problems. For these experiments, three type of image-registration problems were considered: a single-modality case in which two Proton Density magnetic resonance images of a brain were registered (see Figure 3-16), a multi-modality case in which a Proton Density magnetic resonance image of a brain was registered to a T1 magnetic resonance image of the same brain (see Figure 3-17), and a model-to-image registration in which a line model was registered to images of a vertebral endplates (see Figure 3-18). The brain images were taken from the Examples directory of the Insight Toolkit (ITK) [33], a freely available toolkit for image processing, registration and segmentation. The images of the vertebral endplates came from frontal radiographs of scoliosis patients from the database of the scoliosis clinic at Glenrose Rehabilitation Hospital (the images were obtained with approval from the Ethics Panel on Health Research).

The design considerations for these simulations were as follows (See Appendices 2, 4, and 5 for associated computer codes):

- Geometrical transformation. Rigid transformations were used for the model-to-image registration problem. Similarity transformations were used for the single- and multi-modality registration problem. The similarity transformation can be seen as a composition of rotations, translations and uniform scaling.

- Similarity measure. The similarity measure used in the experiments was the Combined Fuzzy Jaccard Index (*CFJI*) as given by equation (3-30) (for the *CFJI* implementation, see Appendix 2). The experiments were also performed with one commonly used similarity measure, the Mean Squared Error (*MSE*) [24], [27], [84] (for the *MSE* implementation, see Appendix 4). Given two overlapping images A and $C=M(B)$, with $M(B)$ being the result of an unknown misalignment M to the image B, the *MSE* is computed by

$$MSE(A,C)=\frac{1}{N}\sum_{x\in\mathbf{X}}|A(x)-C(x)|^2 \qquad (3\text{-}35)$$

with N being the number of pixels that are located in the area of overlap between A and C. *MSE* has a minimum when the images are perfectly aligned and it increases with the misalignment. It is important to note that even though the *MSE* is an image registration similarity measure, it does not satisfy the requirements presented in Section 3.2 for it to be considered a fuzzy sets similarity measure.

- Interpolation method. The linear interpolation method was selected for this work because it provides a good trade-off between computation complexity and interpolation quality.

- Optimization procedure. In this work, a regular step gradient descent optimizer [33] was used. This optimizer is a variant of the gradient descent that attempts to prevent it from taking steps that are too large. At each iteration, the current position ($\mathbf{P}_i$) is computed according to

$$\mathbf{P}_i = \mathbf{P}_{i-1} + \mathbf{S}_i \tag{3-36}$$

with $\mathbf{P}_{i-1}$ being the previous position and $\mathbf{S}_i$ being the current step that is given by

$$\mathbf{S}_i = D_i \mathbf{G}(SM) \tag{3-37}$$

where SM is the value of the similarity measure, $\mathbf{G}(SM)$ is the numerical derivative of the similarity measure, and D_i is the current step length. If the $\mathbf{G}(SM)$ changes of direction, the D_i is given by

$$D_i = D_{i-1} RF \quad (3\text{-}38)$$

with *RF*, the relaxation factor, being a constant in (0, 1) that must be set to a value that prevent the premature shrinkage of the step length in the presence of a noisy metric. For the regular step gradient descent optimizer there are five open parameters that must be set by the user. Two of them (the initial step length, D_o, and the relaxation factor, *RF*) control the speed of convergence while the other three (smallest step length, *Dstop*, smallest gradient magnitude, *Gstop*, and maximum number of iterations, *Niter*) represent stop conditions. In addition to the mentioned five parameters, it is helpful to set scaling factors (or normalization factors) that would put all the variables being optimized in the same range of values. The number of scaling factors would depend on the type of transformation being used. For instance, for a two-dimensional similarity transformation, there would be four scaling factors: two for translations, one for rotation, and one for scaling.

3.4.1 Single-modality registration

For the single-modality experimental part of this study, two proton density magnetic resonance images of a brain were aligned. One proton density magnetic resonance image was used as a reference image (Figure 3-16 (a)). The input image was the result of intentionally rotating the reference image by 10 degrees, scaling it by 0.833 and then translating it by -13 mm in the x-axis and -17 mm in the y-axis (Figure 3-16 (b)). Both images had unit spacing and dimensions of 221 x 257 pixels. The image matching algorithms were implemented using the C++ programming language and the Insight Toolkit (ITK, www.itk.org) [33], a set of open-source set of libraries for medical image processing, registration, and segmentation. The optimizer parameters were set as follows: $D_o = 1$, $RF = 0.5$, $Dstop = 0.0001$, $Gstop = 0.0001$, and $Niter = 500$. The Combined Fuzzy Jaccard Index similarity measure parameters were set as follows: $\alpha = 0.5$ and $\sigma = 1.0$.

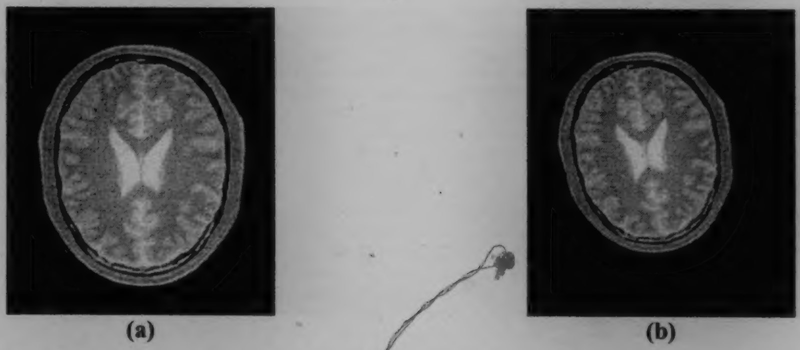

(a) (b)

Figure 3-16: Reference image (a) and input image (b) for the single-modality registration example. The input image is the result of rotating the reference image by 10 degrees, scaling it by 0.833 and then translating it by -13 mm in the x-axis and -17 mm in the y-axis.

3.4.2 Multi-modality registration

For the multi-modality experimental part of this study, two magnetic resonance images of a brain were aligned. A T1 magnetic resonance image was used as a reference image (Figure 3-17 (a)). The input image was a proton density magnetic resonance image that was misaligned by a rotation of 10 degrees, a scaling of 0.833 and a translation of -13 mm in the x-axis and -17 mm in the y-axis (Figure 3-17 (b)). Both images had unit spacing and dimensions of 221 x 257 pixels. The image matching algorithms were implemented using the C++ programming language and the Insight Toolkit (ITK, www.itk.org) [33]. The optimizer parameters were set as follows: $D_o = 1$, $RF = 0.5$, $Dstop = 0.0001$, $Gstop = 0.0001$, and $Niter = 500$. The Combined Fuzzy Jaccard Index similarity measure parameters were set as follows: $\alpha = 0.75$ and $\sigma = 1.0$. The value of α was set to give a higher weight to the gradient magnitudes. This was necessary because for multi-modality images the intensity levels of corresponding anatomical structures may not be in the same range of values.

(a) (b)

Figure 3-17: Reference image (a) and input image (b) for the multi-modality registration example. The input image is rotated 10 degrees, translated -13 mm in the x-axis, translated -17 mm in the y-axis and scaled by 0.833 with respect to the reference image.

3.4.3 Model-to-image registration

For the model-to-image experimental part of this study, a clinical data set from the Scoliosis Clinic at the Glenrose Rehabilitation Hospital in Edmonton, Alberta was used. A frontal radiograph of a patient with scoliosis was used in the experiments. The radiograph was carefully annotated to create the "ground truth". A graphical user interface was used to allow the user to select 16 vertebral endplates to be analyzed. Based on the user selection, regions of interest (ROIs) were created to be used as reference images (see Figure 3-18 (a) for an example of a reference image). A model (Figure 3-18 (b)) was used as input image. Both images had unit spacing and dimensions of 401 x 101 pixels. The image matching algorithms were implemented using the C++ programming language and the Insight Toolkit (ITK, www.itk.org) [33]. The optimizer parameters were set as follows: $RF = 0.5$, $Dstop = 0.0001$, $Gstop = 0.0001$, and $Niter = 500$. To select the appropriate values for the optimizer's parameter D_0 and the *CFJI*'s parameters α and σ, a process of trial and error was carried out.

The candidate sets of parameters were: $D_0 \in \{1,2,\ldots,10\}$, $\alpha \in \{0.25, 0.5, 0.75, 1.0\}$ and $\sigma \in [1,2]$. The goodness of the registration was assessed by visual inspection. The rotation angles obtained with both registration algorithms (using the *CFJI* similarity measure and using the *MSE* similarity measure) were compared with those measured by an expert to compute the precision of each registration algorithm.

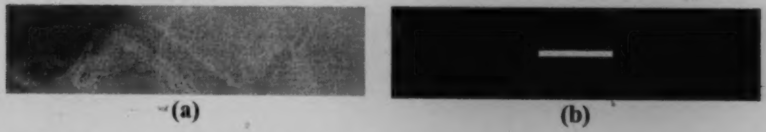

(a) (b)

Figure 3-18: Sample reference image (a) and model (b) for the model-image registration example.

3.5 Results

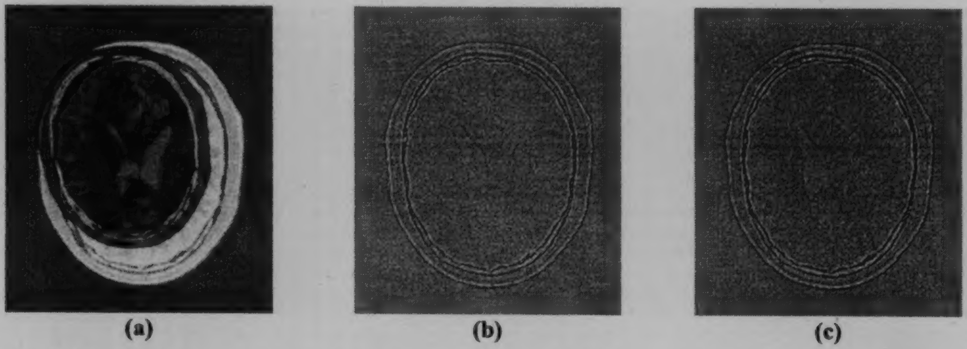
3.5.1 Single-modality registration problem

This section compares the results obtained using the *CFJI* and *MSE* similarity measures for a single-modality registration case. The results obtained using both similarity measures were close to each other and close to the expected results (Table 3-3). Using the *CFJI* similarity measure, the registration took 71 optimization iterations and the resulting similarity measure value was 0.9. Using the *MSE* similarity measure, the registration took 42 optimization iterations and resulting similarity measure value was 52.96.

	Expected	Obtained	
		CFJI	*MSE*
Scale Factor	0.833	0.833	0.833
Rotation Angle (°)	10	10.0	10.0
Translation in X (mm)	13	13.3	13.3
Translation in Y (mm)	17	15.8	15.7

Table 3-3: The resulting transformation parameters for registration of Proton Density Magnetic Resonance Images of a brain, using the *CFJI* and *MSE* similarity measures.

Figure 3-19 (a) shows the squared magnitude of pixel intensity values differences between the reference image and the input image before registration. Figure 3-19 (b) shows the squared magnitude of pixel intensity values differences between the reference image and the resampled input image after registration done using the *CFJI* similarity measure. Figure 3-19 (c) shows the squared magnitude of pixel intensity values differences between the reference image and the resampled input image after registration done using the *MSE* similarity measure. There is no visually detectable difference between the outputs obtained using the *CFJI* similarity measure and the output obtained using the *MSE* similarity measure.



(a) (b) (c)

Figure 3-19: Difference before registration (a) and difference after registration using the *CFJI* (b) and *MSE* (c) similarity measures for the single-modality registration example.

3.5.2 Multi-modality registration problem

This section compares the results obtained using the *CFJI* and *MSE* similarity measures for a multi-modality registration case. The results obtained using the *CFJI* similarity measure were close to the expected results (Table 3-4). From the results obtained using the *MSE* similarity measure, the translations were close to the expected results (Table 3-4) but the rotation and the scaling were not. Using the *CFJI* similarity measure, the registration took optimization 500 iterations and the resulting similarity measure value was 0.5. Using the *MSE* similarity measure, the registration took 57 optimization iterations and resulting similarity measure value was 3107.49.

	Expected	Obtained	
		CFJI	*MSE*
Scale Factor	0.833	0.845	0.969
Rotation Angle (°)	10	10.1	27.4
Translation in X (mm)	13	13.9	12.9
Translation in Y (mm)	17	15.4	16.2

Table 3-4: The resulting transformation parameters for the registration of a Proton Density Magnetic Resonance Image to a T1 Magnetic Resonance Image of a brain, using the *CFJI* and *MSE* similarity measures.

Figure 3-20(a) shows the squared magnitude of pixel intensity values differences between the reference image and the input image before registration. Figure 3-20(b) shows the squared magnitude of pixel intensity values differences between the reference image and the resampled input image after registration done using the *CFJI* similarity measure. Figure 3-20(c) shows the squared magnitude of pixel intensity values differences between the reference image and the resampled input image after registration done using the *MSE* similarity measure. The difference after registration done using the *MSE* similarity measure shows that the images were not properly registered.

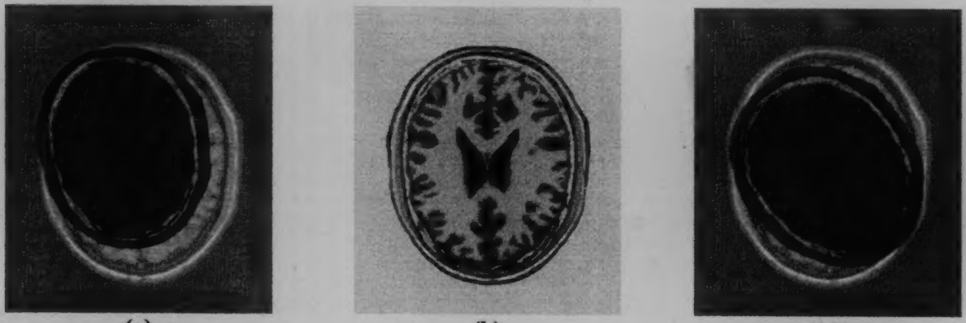

(a) **(b)** **(c)**

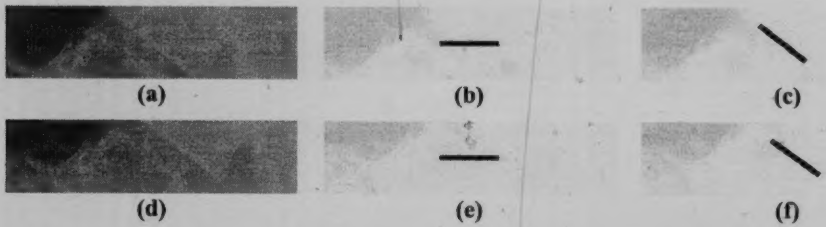
Figure 3-20: Difference before registration (a) and difference after registration using the *CFJI* (b) and *MSE* (c) similarity measures for the multi-modality registration example.

3.5.3 Model-to-image registration problem

This section compares the results obtained using the *CFJI* and *MSE* similarity measures for a model-to-image registration case. After performing visual inspection to determine the goodness of the registration solutions, it was determined that in 100% of the cases (16 out of 16), the results obtained using the *CFJI* similarity measure were acceptable. On the other hand, the results obtained with the *MSE* similarity measure were acceptable in 80% of the cases (12 out of 16). The rotation angles obtained using the *CFJI* similarity measure showed a very high correlation (0.99) to the expected results (Table 3-5). Moreover, the absolute registration error for the rotation angle was only 1.2° in the case of the *CFJI* similarity measure. From the results obtained using the *MSE* similarity measure, it can be seen that even though the registration failed in four cases, the obtained rotation angle showed a high correlation (0.76) to the expected results.

Figure 3-21 shows two examples of registration solutions obtained using the *CFJI* similarity measure. Figure 3-21(a) shows the reference image. Figure 3-21(b) shows the squared magnitude of pixel intensity values differences between a reference image and a model before registration. Figure 3-21(c) shows the squared magnitude of pixel intensity values differences between the reference image and the resampled model after registration. Figure 3-21(d) shows the reference image. Figure 3-21(e) shows the squared magnitude of pixel intensity values differences between another reference image and a model before registration. Figure 3-21(f) shows the squared magnitude of pixel intensity values differences between the reference image and the resampled model after registration.

Figure 3-22 shows the four unacceptable registration solutions obtained using the *MSE* similarity measure. Figure 3-23 shows the corresponding four registration solutions obtained using the *CFJI* similarity measure.


(a) **(b)** **(c)**

(d) **(e)** **(f)**

Figure 3-21: Reference image (a), difference before registration (b) and difference after registration (c) for model-to-image registration of a bottom vertebral endplate using the *CFJI* similarity measure. Reference image (d), difference before registration (e) and difference after registration (f) for model-to-image registration of a top vertebral endplate using the *CFJI* similarity measure.

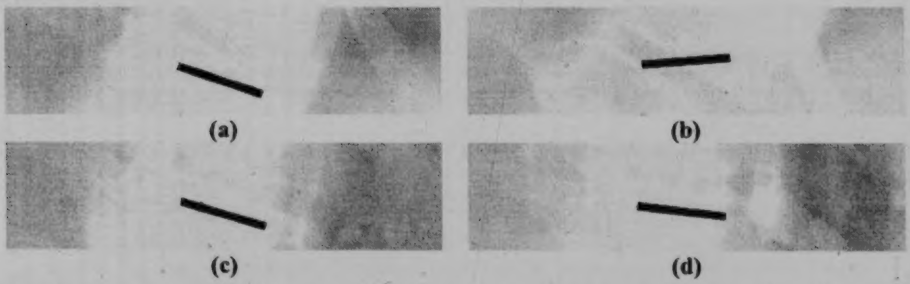


Figure 3-22: The four bad registration solutions out of 16 obtained using the *MSE*. (a) Registration error of 11° when matching a top endplate. (b) Registration error of 39° when matching a bottom endplate. (c) Registration error of 5° when matching a top endplate (note that the model was not translated correctly). (d) Registration error of 2° when matching a top endplate (note that the model was not translated correctly).

Figure 3-23: The registration solutions obtained with the *CFJI* similarity measure that correspond to the four bad registration solutions out of 16 obtained using the *MSE*. (a) Registration error of 1° when matching a top endplate. (b) Registration error of 0° when matching a bottom endplate. (c) Registration error of 1° when matching a top endplate. (d) Registration error of 1° when matching a top endplate.

	CFJI	*MSE*
Percentage of successful registrations	100 %	80%
Absolute error in rotation angle (mean ± stdev)	1.2° ± 0.9°	5.3° ± 9.3°
Absolute error in rotation angle ([min, max])	[0.2°, 2.9°]	[0.4°, 38.4°]
Correlation coefficient (expected vs obtained rotation angle)	0.99	0.76

Table 3-5: Registration performance results for the model-to-image registration problem using the *CFJI* and *MSE* similarity measures.

3.6 Discussion

A fuzzy computing algorithm was developed for the registration of fuzzy sets representations of intensity values and gradient magnitude values. The key component of the proposed algorithm is a similarity measure based on the fuzzy Jaccard index -the Combined Fuzzy Jaccard Index (*CFJI*). The *CFJI* allows focusing in intensity values, gradient magnitudes, or both to better deal with a variety of registration problems. In the registration cases presented in this chapter, the proposed approach outperformed a *MSE*-based registration algorithm in terms of precision and percentage of good registration solutions in multimodal and model-to-image registration problems.

The studies presented in the previous section indicated that the current implementation of the *CFJI*-based registration algorithm performed well in the single-modality, multimodality, and model-to-image registration problems. One important feature of the Combined Fuzzy Jaccard Index similarity measure is the presence of the configuration parameters α and σ that can be used to control the behaviour of the similarity measure in attention to the registration problem at hand. For single-modality registration problems, $\alpha = 0.5$ is usually adequate to get a good registration result. In more challenging scenarios, such as the registration of images of different modalities or the registration of models to images, it is necessary to adjust the value of α until acceptable results are obtained. In practice, the similarity measure parameters must be adjusted to suit the problem at hand. Specifically, adapting α and σ to the type of images or noise level is recommended. For instance, for weak edges low values of α and σ are commended. $\sigma \in [1, 2]$ usually performs well in practice. Regarding the value of α, values in $\{0, 0.25, 0.5, 0.75, 1.0\}$ are adequate for a variety of scenarios. It is important to note that for α=0, the value of σ has no effect on the response of the similarity measure to various levels of misalignment. This is due to the fact that the value of σ is only used in the computation of FJI_g. For α=0, the value of FJI_g is not taken into account in the computation of the *CFJI*.

The *CFJI* is a good similarity measure for registration problems dealing with images with variable contrast and moderate focus such as the clinical images associated with scoliosis management. However, for images that are well-focused, are of high resolution, and are of only one modality, similarity measures like the *MSE* may be preferred, in particular, due to its low computational cost. In all the previous simulations, the *MSE*-based registration algorithm was about three times faster than the *CFJI*-based registration algorithm (which took an average of 1 second per optimization iteration for the model-to-image registration problem).

In short, the proposed approach is a good alternative for medical tasks involving the matching of models to images.

3.7 Summary

This chapter has described a novel approach to use fuzzy similarity measures for image registration. The theory behind such approach was presented and some examples illustrating its features were described. Three experiments were presented to show the usefulness of the proposed similarity measure in a registration algorithm and to compare its performance to that of a Mean Squared Errors-base registration algorithm.

The next chapter will present a computational intelligence approach to estimate the quality of fit of an image registration solution.

4 A support vector machines classifier to assess the quality of registration solutions from fuzzy overlap measures[4]

4.1 Introduction

Medical image registration [24], [27], [84] is a key component of image analysis applications. It is used for extracting clinical information for the diagnosis of medical conditions and for the planning and evaluating of therapeutic procedures. Before extracting clinical information from aligned images, it is necessary to verify the quality of the registration solution. Bad registration solutions may lead to uncertainties that jeopardize the validity of the diagnosis and/or the validity of the planning and evaluating of therapeutic procedures. In this chapter, a novel application of measures of fuzzy sets overlap is presented. The application consists on automatically determining the quality of registration results. Because the clinical focus of this work was on spinal deformities, the main interest was on measuring inclinations of vertebrae. To measure such angles a model-to-image registration approach was used. The model-to-image registration approach involved registering a model (in the form of a line) to vertebral endplates to estimate the tilt angle of the endplates. A Support Vector Classifier (SVC) [76], [77], [47] was used to find a relation between the proposed overlap measure and the quality of the registration results as assessed by an expert. The main goal was to determine whether a SVC, using the proposed overlap measure, could predict the quality of the registration results sufficiently well to be used in clinical practice. A SVC was chosen because, unlike techniques such as Artificial Neural Networks [3], the support vector machines theory offers the possibility to train generalizable non-linear classifiers using small datasets as is usually the case in scoliosis research. Moreover, unpublished preliminary studies comparing the performance of radial basis function neural networks [3] and SVC indicated the superiority of the latter in the datasets considered in this work. Finally, the results of applying a SVC to the dataset of scoliosis patients was compared to those obtained by applying logistic regression [31] and classification decision trees [12] to the dataset to fulfill the requirements of comparability needed for decision support systems [65].

---

[4] A version of this chapter has been submitted for publication. L. Ramirez, N. Durdle, J. Raso, IEEE Transactions on Information Technology in Biomedicine. 2007.

4.2 Proposed overlap measure

Two-dimensional digital images can be identified with fuzzy sets that take values on the grid points (*i*, *j*). Using the notation of fuzzy sets [82], one can write the fuzzy set representation of a digital image as an *M* x *N* array,

$$A = \begin{bmatrix} A(x_{11}) & A(x_{12}) & \cdots & A(x_{1j}) & \cdots & A(x_{1N}) \\ A(x_{21}) & A(x_{22}) & \cdots & A(x_{2j}) & \cdots & A(x_{2N}) \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ A(x_{i1}) & A(x_{i2}) & \cdots & A(x_{ij}) & \cdots & A(x_{iN}) \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ A(x_{M1}) & A(x_{M2}) & \cdots & A(x_{Mj}) & \cdots & A(x_{MN}) \end{bmatrix} \qquad (4\text{-}1)$$

where $i, j \in \aleph$, $0 \le i \le M$ and $0 \le j \le N$ with M and N being the dimensions of the digital image. $A(x_{ij})$ ($0 \le A(x_{ij}) \le 1$) represents the membership value of the (*i*,*j*)th pixel to a fuzzy set. By representing a digital image as a fuzzy set, one can use all the operations on fuzzy sets in the analysis of images. In this chapter, the interest is on operations that allow measuring the similarity between fuzzy sets. The interest in the similarity measures comes from the following observation: two medical images to be registered are related through the common anatomical structure they measure. Therefore, when images are correctly registered, the corresponding structures overlap. When the overlap is maximal, the similarity between the fuzzy set representations of the images is maximal. Based on that observation, one can use fuzzy similarity measures to assess the quality of the registration results. In this work, the focus will be on the Fuzzy Jaccard Index (*FJI*) [16]:

$$FJI(A,C) = \frac{\sum_{x\in \mathbf{X}} min(A_i(x),C_i(x))}{\sum_{x\in \mathbf{X}} max(A_i(x),C_i(x))} \qquad (4\text{-}2)$$

with A being the fuzzy set representation of the reference image and C being the fuzzy set representation of the registered input image.

The *FJI* is a generalization of the Jaccard Index (or Tannimoto Coefficient) [15]. The Jaccard Index has been successfully used for the evaluation of image segmentation applications [15], [38], [74], for the comparison of images to evaluate image processing applications [16], and for controlling image registration processes [15]. Given two overlapping images A and $C = M(B)$, with $M(B)$ being the result of an unknown misalignment M to the image B, the Jaccard Index (*JI*) is computed by

$$JI(A,C) = \frac{|A \cap C|}{|A \cup C|} \qquad (4\text{-}3)$$

$|A \cap C|$ is the number of elements in the intersection, which, in the case of the *FJI*, is modeled by $\sum_{x\in \mathbf{X}} min(A_i(x),C_i(x))$. $|A \cup C|$ is the number of elements in the union, which, in the case of the *FJI*, is modeled by $\sum_{x\in \mathbf{X}} max(A_i(x),C_i(x))$.

The *JI* requires knowledge of the labels identifying elements in the regions of interest. These labels are not always available. To deal with this situation, the use of a fuzzy sets representation of image features is proposed. By using fuzzy sets instead of labels, one can use the *JI* while effectively exploiting the available information in the images. The main features considered are the edges of the images because they contain useful structural information about the boundaries of the objects of interest. In particular, the focus is on the gradient magnitude that measures the local steepness of the intensity landscape, which has local maxima at the edges. With this in mind, one can use the *FJI* on gradient magnitude (*FJIG*) to quantify the fuzzy overlap between fuzzy set representations of the edges in both images. The aim of the *FJIG* is to exploit the functional relation between sets of pixels having large gradient magnitudes in both images.

The *FJIG* measures the amount of area overlap between input and reference images with respect to the total area. This is precisely how a domain expert would evaluate the quality of the registration solution. However, the numeric value of the *FJIG* cannot be used directly as a measure of the quality of the registration solution because the value of the *FJIG* is affected not only by the amount of misalignment but also by the type of registration problem being solved (single-modality, multimodality, or model-to-image registration) and the scale factor (σ) used in the computation of the gradient. To deal with this situation, a Support Vector Classifier (SVC) will be used to find a relation between the *FJIG* and the quality of the registration results as assessed by an expert.

4.3 Proposed use of the system

The proposed overlap measure is used to evaluate the quality of registration solutions in a clinical application for measuring the tilt angle of vertebral endplates in spine radiographs of patients with scoliosis. The proposed use of the system is as follows (see Appendix 1 for a general overview on the use of the system). A graphical user interface, written in the Perl programming language, was used to allow the user to select the vertebral endplates to be analyzed. Figure 4-1 shows an example of graphical user interface for the measurement of vertebral wedging. Figure 4-2 shows an example of graphical user interface for the measurement of Cobb angle. Based on the user selection, regions of interest (ROIs) were created (Figure 4-3(a)). A model (Figure 4-3(b)) was then registered to the ROIs, using custom-made software, to find the location and tilt angles of the vertebral endplates under study. The software used is depicted in Appendix 5. If the registration solution indicates a tilt angle greater than 45°, the solution was flagged as requiring human assessment because such a tilt suggests either a large misregistration or a very high Cobb angle. For registration solutions with a tilt angle of less than 45°, a two-class classifier was used to discriminate groups of registration solutions based on their tilt angle: good registration solutions (in which the difference between the obtained tilt angle and the expected tilt angle was less than or equal to 3°) and bad registration solutions (in which the difference between the obtained tilt angle and the expected tilt angle was greater than 3°). Good registration solutions are used to compute the vertebral wedging or Cobb angle and the results are shown graphically as indicated in Figure 4-4 and Figure 4-5.

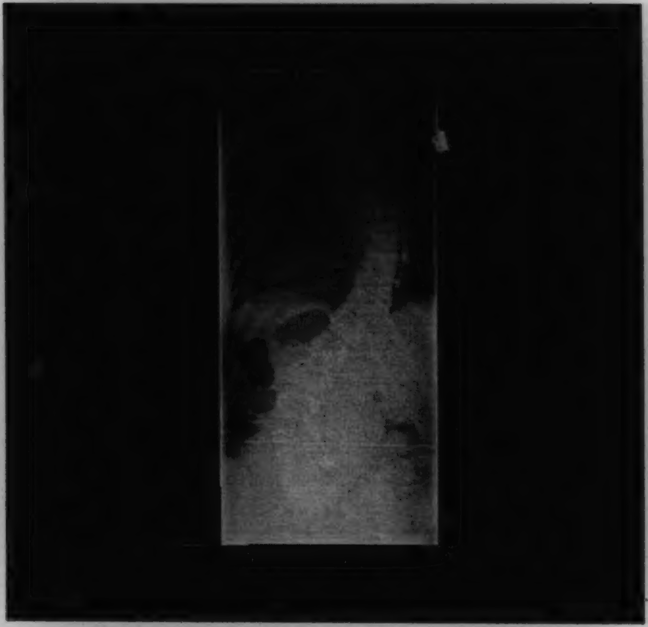


Figure 4-1: Graphical user interface used to select vertebral endplates for the measurement of vertebral wedging.

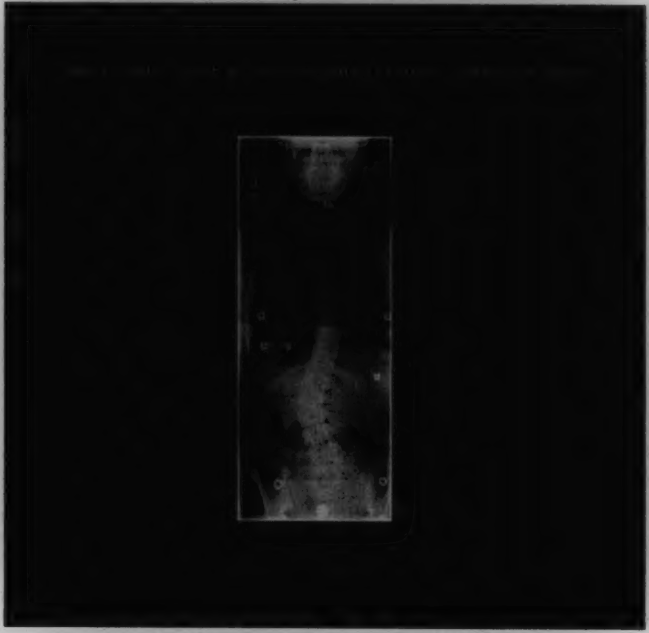
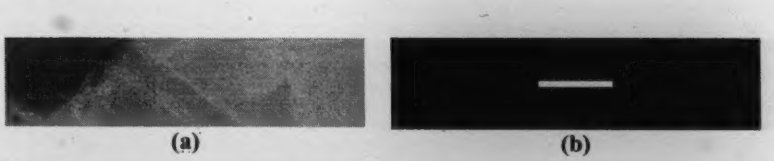


Figure 4-2: Graphical user interface used to select vertebral endplates for the measurement of Cobb angle.



(a) (b)

Figure 4-3: Sample reference image (a) and model (b) for model-to-image registration.

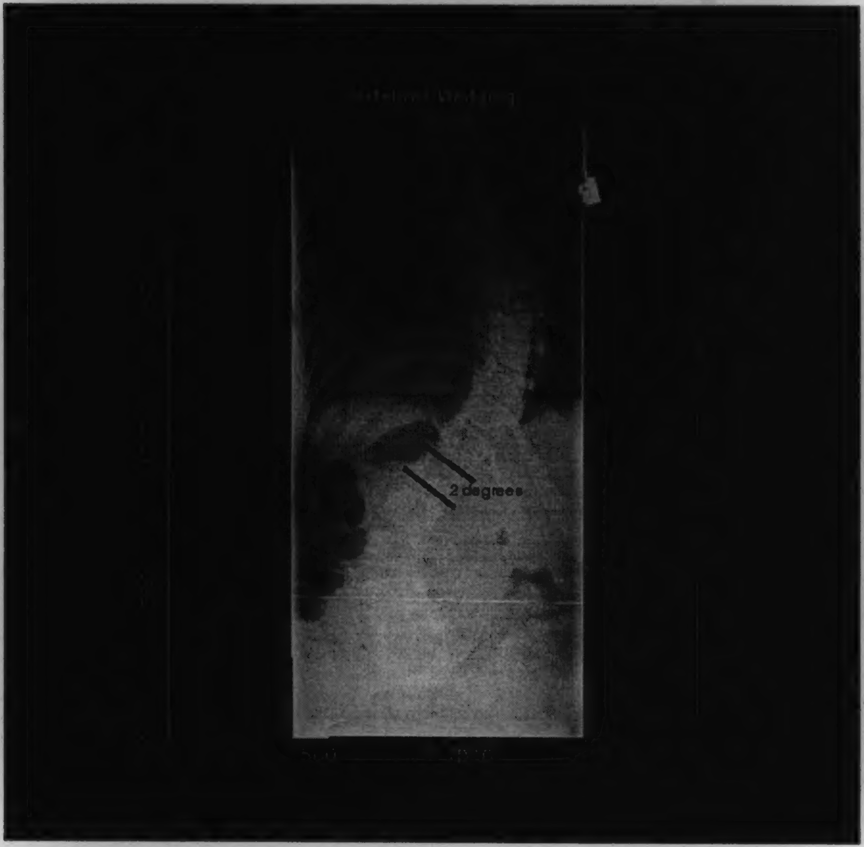


Figure 4-4: Measured wedge for vertebra under study.

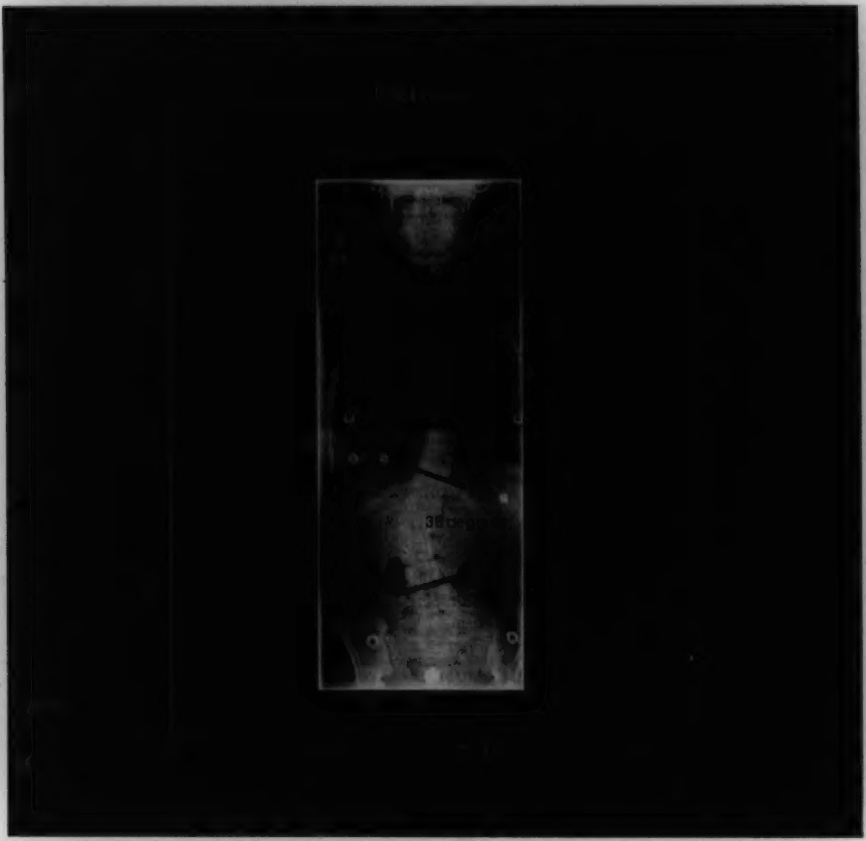


Figure 4-5: Measured Cobb angle.

4.4 Patient data set

Retrospectively, radiographs of patients with scoliosis from the database of the scoliosis clinic at the Glenrose Rehabilitation Hospital were examined, after getting Ethics Approval, to select patients for the study. The following inclusion criteria were used: 1) having available a posterior-anterior standing radiograph with a maximum Cobb angle of less than 75°; 2) not having undergone surgery; and 3) having at least a visually identifiable vertebral endplate. Eighteen patients and a total of 141 vertebral endplates satisfied the inclusion criteria. Figure 4-6 shows an example of a region of interest around a vertebral endplate that satisfied the inclusion criteria. Patients had a variety of spinal curvatures, with an average of maximum

Cobb angle of 39° ± 18° (range 8°-74°). There were 7 (39%) patients with double curve and 11 (61%) patients with single curves.

After performing registration on the ROIs corresponding to the 141 vertebral endplates under study, the *FJIGs* on the ROIs and the models were computed (see Appendix 6). In addition, the mean value ($MeanG$), median value ($MedianG$), range of values ($RangeG$), and standard deviation ($StDevG$) of the gradient magnitudes of the ROIs were computed by (see Appendix 6)

$$MeanG(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \tag{4-4}$$

with n being the number of samples $\mathbf{x}_i$. in the data set.

$$MedianG(\mathbf{x}) = 50^{th}\ \text{percentile}(\mathbf{x}) \tag{4-5}$$

$$RangeG(\mathbf{x}) = max(\mathbf{x}) - min(\mathbf{x}) \tag{4-6}$$

$$StDevG(\mathbf{x}) = \sqrt{\left(\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x} - MeanG(\mathbf{x}))^2\right)} \tag{4-7}$$

with n being the number of samples $\mathbf{x}_i$. in the data set.

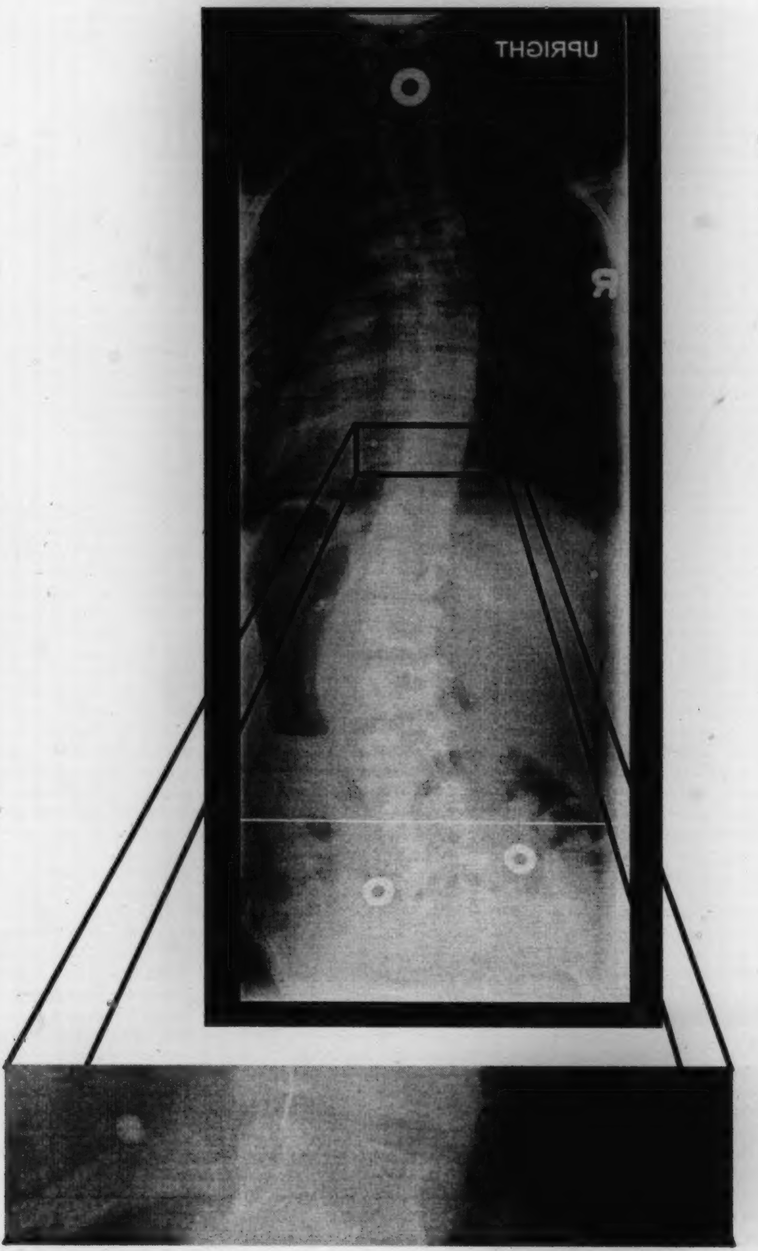
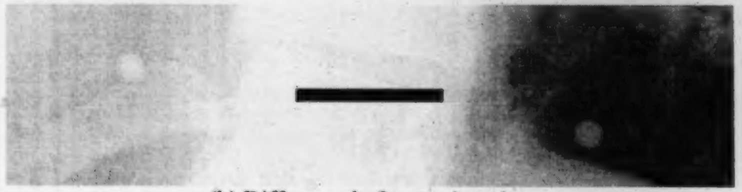


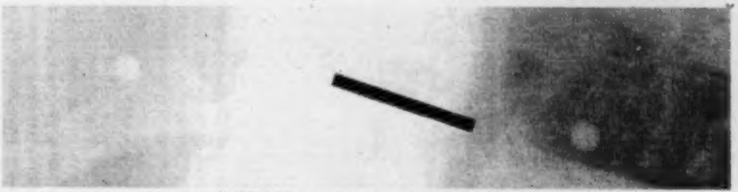
Figure 4-6: Example of a vertebral endplate that satisfied the inclusion criteria.

It is important to note that the lack of a known transformation makes evaluation of registration accuracy difficult. To overcome this challenge, a set of landmarks to be used as a ground truth for the evaluation of the registration algorithms were defined. To define the ground truth, each of the 18 spine radiographs was rendered and selected landmarks were traced manually using anatomic criteria. The landmarks chosen were lines going through the most visible edge of each vertebral endplate. The tilt angle for each line was also recorded. The landmarks were traced before reviewing the registration results. Moreover, the same landmarks were used to evaluate all the registration results obtained with the different algorithms. This approach ensures that any inaccuracies in landmark identification are not biased with respect to the model or method being tested and that such inaccuracy is equally likely to be detrimental to the apparent performance of any of the registration strategies. To indirectly assess the accuracy of the location of the landmarks, a study was performed to determine the intra- and inter-observer reliability of the Cobb angle measurements made by the person that identified the landmarks and by a well-experienced orthopaedic surgeon. Twenty-three Cobb angles were used in this study. The measurements were carried out independently three times with a three week interval between sessions. The intra-observer variability was 2° ± 1° (with a range of 0° - 5°). The inter-observer variability was 3°± 2° (with a range of 0° - 10°). These results agree with the accepted intra- and inter-observer variability of 4° to 8° [23] suggesting that, in fact, the measurements made can serve as gold standard.

(a) ROI of top vertebral endplate under study



(b) Difference before registration



(c) Difference after registration

Figure 4-7: Sample reference image used in the model-to-image registration example (a). Difference before registration (b) and difference after registration (c) for a model-to-image registration example of a top vertebral endplate using the *CFJI* similarity measure.

4.5 Support vector classifiers

Support vector classifiers (SVC) [77], originally designed to solve two-class classification problems, have been used with a high degree of success in several applications ranging from bioinformatics to text categorization. In their basic form, SVC learn linear decision rules of the following form:

$$D(\mathbf{x}) = sign((\mathbf{w} \cdot \mathbf{x}) + b) \tag{4-8}$$

The decision rule $D(\mathbf{x})$ is equal to +1 for positive values of $(\mathbf{w} \cdot \mathbf{x}) + b$. $D(\mathbf{x})$ is equal to -1 for negative values of $(\mathbf{w} \cdot \mathbf{x}) + b$. The weight vector $\mathbf{w}$ and threshold b are found during training and they describe a hyperplane (a higher-dimensional generalization of a plane in 3-

dimensional Euclidean geometry). Observations (**x**) are classified according to the side of the hyperplane in which they are located. During training, the SVC approach finds the hyperplane that classifies most training samples correctly while keeping the largest separation possible between classes. If all the training samples can be classified without error, the distance from the hyperplane to the closest training sample is known as the margin. Otherwise, the margin is the distance from the hyperplane to the closest correctly classified samples.

For cases in which the data set cannot be correctly classified by a linear decision rule, the training observations (**x**) are mapped, by a function $\Phi(\mathbf{x})$, to a feature space where the observations become linearly separable. The resulting decision rule has the form:

$$D(\mathbf{x}) = sign((\mathbf{w} \cdot \Phi(\mathbf{x})) + b) \tag{4-9}$$

w and b are found during training. To find the **w** that maximizes the margin, a dual optimization problem is usually solved, leading to the following decision rule (the reader is referred to [47] for a detailed derivation):

$$D(\mathbf{x}) = sign\left(\sum_{i=1}^{n} \beta_i \lambda_i (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x})) + b\right) \tag{4-10}$$

with n being the number of samples in the training set.

This decision rule is often written as

$$D(\mathbf{x}) = sign\left(\sum_{i=1}^{n} \beta_i \lambda_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \tag{4-11}$$

with n being the number of samples in the training set and with $K(\mathbf{x}_i,\mathbf{x})$ being a kernel function of a pattern to be classified $\mathbf{x}$ and a training pattern $\mathbf{x}_i$. Examples of kernel functions are a linear kernel ($K(\mathbf{x}_i,\cdot\mathbf{x}) = (\mathbf{x}_i \cdot \mathbf{x})$) and a radial basis function kernel ($K(\mathbf{x}_i,\cdot\mathbf{x}) = exp\left(-\gamma\|\mathbf{x}_i \cdot \mathbf{x}\|^2\right), \gamma > 0$). S is a subset of the training set. $\lambda_i \in \{-1, 1\}$ is the label of sample $\mathbf{x}_i$. β_i is the Lagrange multiplier associated with the sample $\mathbf{x}_i$. b is the threshold. As described in [77] and [47], during training, optimization of β_i is achieved by

$$max_\beta\left(\sum_{i=1}^{n}\beta_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\beta_j\lambda_i\lambda_j K(\mathbf{x}_i,\mathbf{x}_j)\right) \tag{4-12}$$

Subject to

$$\sum_{i=1}^{n}\beta_i\lambda_i = 0 \tag{4-13}$$

and

$$0 \le \beta_i \le C \tag{4-14}$$

The class overlap is controlled by the penalty weight $C > 0$. For $C \to \infty$, no overlap is allowed. During optimization, the values of all β_i become 0, except for the β_i associated with the support vectors (a special subset of the training samples). Consequently the support vectors are the only patterns needed in deciding the position of the decision boundary.

Summarizing the SVC approach, the input vector $\mathbf{x}$ and the support vectors $\mathbf{x}_i$ are mapped into a feature space where the necessary dot products are computed by using the kernel function K. The kernel function is chosen a priori, and it determines the type of classifier (e.g., linear or radial basis function). The penalty weight C is also chosen a priori. C controls the amount of allowed overlap between classes. All other parameters (the number of support vectors and weights and the threshold) are found during training.

4.6 Logistic regression

Linear regression is used to explore the relation between a response variable and one or more explanatory variables. However, when the response variable is binary, linear regression is not appropriate because in linear regression results are, in general, unbounded as the explanatory variables tend to infinity and with a binary response variable there are only two possible values [31]. Examples of binary response variables are: large vs. Small misregistrations, presence vs. absence of a condition, occurrence vs. non-occurrence of a response within a defined period of observation. For these types of examples, the logistic regression (LR) [31] model has become one of the most commonly used methods of analysis [31]. Logistic regression assumes that the relation between the explanatory variables and the response variable is nonlinear. In logistic regression, the response variable is given by

$$D(\mathbf{x}) = \frac{exp\left(\beta_0 + \sum_{\forall x_i \in \mathbf{x}} \beta_i x_i\right)}{1 + exp\left(\beta_0 + \sum_{\forall x_i \in \mathbf{x}} \beta_i x_i\right)} + \varepsilon \qquad (4\text{-}15)$$

where β_i are the regression coefficients; x_i are the explanatory variables in $\mathbf{x}$; and ε is a random disturbance term that is assumed to be uncorrelated to the explanatory variables x_i [31].

4.7 Decision trees

Decision trees (DT) [12] consist of nodes and branches connecting the nodes. The nodes located at the bottom of the decision tree are called leaves. They indicate the classes associated with the response variables. The top node in the decision tree is called the root. The root contains all the training samples associated with the explanatory variables. All nodes but the leaves are called decision nodes because they specify some decision to partition the data set based on a selected explanatory variable. Figure 4-8 shows a decision tree used for the classification of three types of flowers. In the figure, it can be seen that there are five leaves nodes (one node for the class iris-setosa and two nodes for each of the remaining classes). Decision trees are classifiers trained by an iterative process in which features are selected and corresponding thresholds are identified at each node of the tree to improve the prediction accuracy. The main advantages of decision trees are [35]:

- their speed (because only the explanatory variables needed to classify the test pattern of interest are considered, the classification can be done with a low computational cost);
- their interpretability (because decision trees can be represented by a group of if then rules in terms of the selected explanatory variables, the results are clearly interpretable); and
- their capability to handle categorical and numerical data.

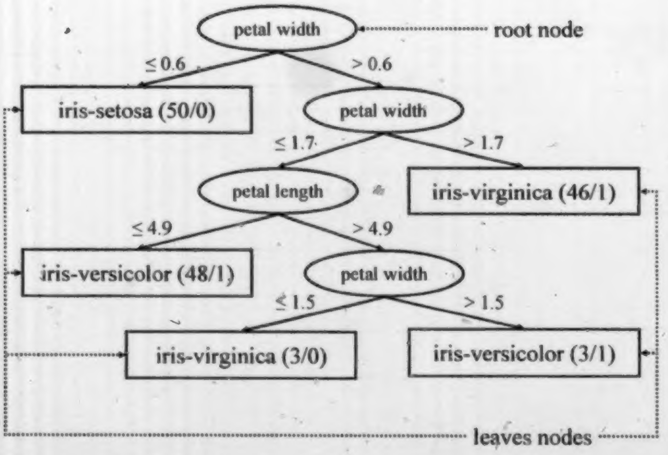

petal width
root node
≤ 0.6
> 0.6
iris-setosa (50/0)
petal width
≤ 1.7
> 1.7
petal length
iris-virginica (46/1)
≤ 4.9
> 4.9
iris-versicolor (48/1)
petal width
≤ 1.5
> 1.5
iris-virginica (3/0)
iris-versicolor (3/1)
leaves nodes

Figure 4-8: A decision tree for the classification of three types of flowers: iris-setosa, iris-versicolor, and iris-virginica. The leaves nodes indicate the number of samples in each particular class (e.g. the leaf node iris-setosa classified correctly 50 samples while 0 samples were incorrectly classified). The values in the branches indicate the criteria to divide the data set (e.g. if petal width is less than or equal to 0.6 then the flower is iris-setosa).

4.8 Data preparation

The data set for the experiments was divided into training and test set according to the time in which the data became available. The training set consisted of 14 radiographs from which 104 vertebral endplates were selected using the selection criteria previously discussed. The test set consisted of 4 radiographs that became available after the first group. From the second group of radiographs, 37 vertebral endplates were selected following the inclusion criteria previously discussed. Once the endplates were selected, model-to-image registrations were performed using the proposed similarity measure (*CFJI*) described on Chapter 3 (see Appendices 2, 4, and 5 for the algorithms and computer codes). The parameters used for the combined fuzzy Jaccard index similarity measure were $\alpha = 0.5$ and $\sigma = 2.0$. The optimizer parameters were set to $RF = 0.5$, $D_0 = 5.0$, $Dstop = 0.0001$, $Gstop = 0.0001$, and $Niter = 500$. After registration, each solution was evaluated with respect to the ground truth. If the difference in the rotation angle was greater than 3° the solution was labelled as a Large Misregistration (LM). Otherwise, the location of the registered model was assessed visually to assign one of two possible labels: Large Misregistration (LM) or Small Misregistration (SM). For the training set there were 54 registration solutions labelled as LM and 50 registration solutions labelled as SM. For test set there were 25 registration solutions labelled as LM and 12 registration solutions labelled as SM. It is important to note that these registration results were obtained without any optimization of the registration parameters. No optimization was attempted because for training and testing the classifiers, it was necessary to have good registration solutions and bad registrations solutions. Therefore, the criterion for setting up the parameters was to select those parameters that provided the greater number of good registration in Chapter 3. After registration, the overlap measures presented in this chapter were computed. To compute the gradient images $\sigma = 1.0$ was used.

4.9 Exploratory data analysis

Table 4-1 depicts statistical information of the features contained in the training data set. From Table 4-1, one can notice that all the explanatory variables had a significant correlation to the class of registration solutions (Large Misregistration and Small Misregistrations). The *MeanG* and *MedianG* had similar mean value (0.363 for *MeanG* and 0.349 for *MedianG*) and standard deviation (0.203 for *MeanG* and 0.204 for *MedianG*). Based on this observation one could hypothesize that the *MeanG* and *MedianG* were related. To test that hypothesis, the correlation coefficient between *MeanG* and *MedianG* was computed. The correlation coefficient between *MeanG* and *MedianG* was $r = 0.99$ suggesting that there was a very large linear correlation between the two variables. The coefficient of determination was $r^2 = 0.98$ suggesting that 98% of the total variation in one variable could be expressed by the linear relation between both variables. These findings suggest that the hypothesis that *MeanG* and *MedianG* were related was true. Because of the very high correlation between *MeanG* and *MedianG*, it was possible to assume that they were redundant. Therefore, for the classification problem only the variable with the highest correlation to the class of registration solutions (*MeanG*) was considered.

Figure 4-9 through Figure 4-14 provide a graphical representation of the training set variables used in the classification experiments. Patterns associated with the class of Large Misregistrations were identified with crosses while patterns associated with the class of Small Misregistrations were identified with dots. Figure 4-9 through Figure 4-11 show samples belonging to the same class grouped together. This may suggest that *FJIG* in combination with any of the other explanatory variables have a good discriminatory power of the class of registration solutions. Figure 4-12 through Figure 4-14 show samples belonging to the same class scattered over the graph. This may suggest that, compared to using *FJIG* with other explanatory variables, it is harder to discriminate the class of registration solutions by using a combination of the explanatory variables *MeanG*, *RangeG*, and *StDevG*.

Name	Mean	SD	Min	Max	r
FJIG	0.020	0.005	0.009	0.035	0.781*
MeanG	0.363	0.203	0.000	1.000	0.579*
MedianG	0.349	0.204	0.000	1.000	0.550*
RangeG	0.544	0.266	0.000	1.000	0.573*
StDevG	0.429	0.228	0.000	1.000	0.591*

Table 4-1: List of features of the training set. SD: standard deviation. N: Number of vertebral endplates = 104. r: Pearson coefficient of correlation to the class of registration solutions; for $N=104$, $r>0.193$ for significance at $p<0.05$. * indicates significance at $p<0.05$.

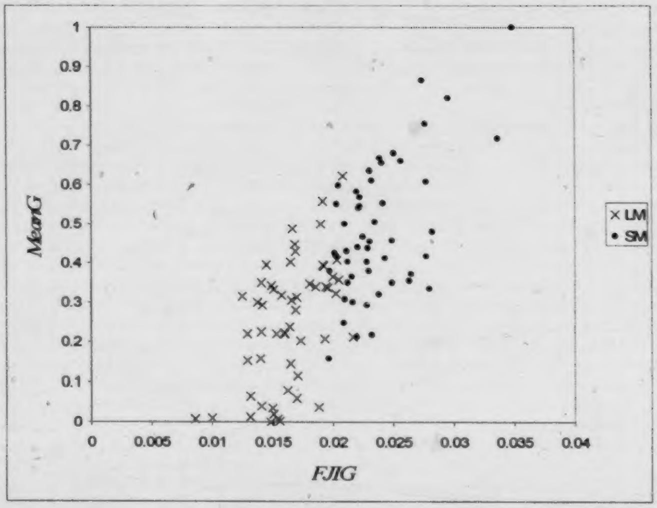


Figure 4-9: Scatter plot of *FJIG* versus *MeanG* for the training set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

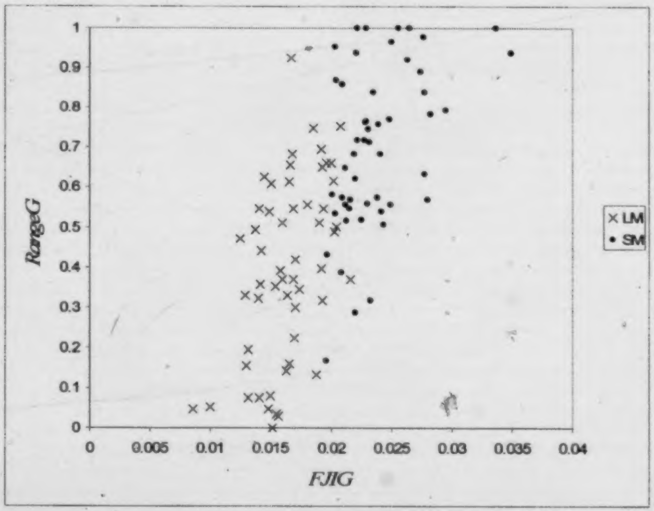


Figure 4-10: Scatter plot of *FJIG* versus *RangeG* for the training set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

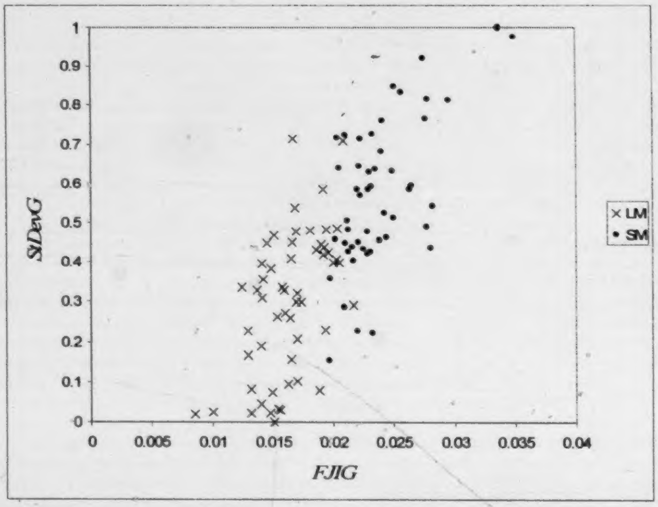


Figure 4-11: Scatter plot of *FJIG* versus *StDevG* for the training set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

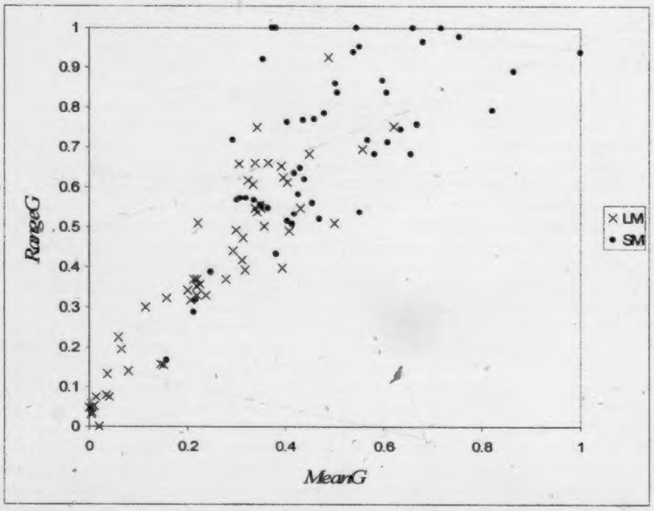


Figure 4-12: Scatter plot of *MeanG* versus *RangeG* for the training set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

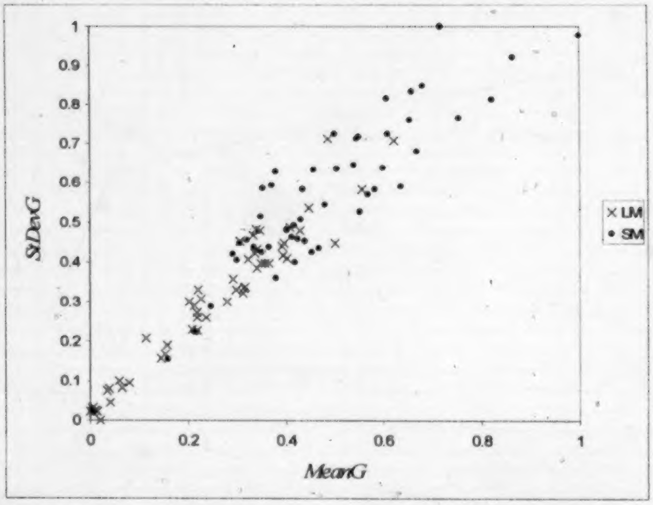


Figure 4-13: Scatter plot of *MeanG* versus *StDevG* for the training set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

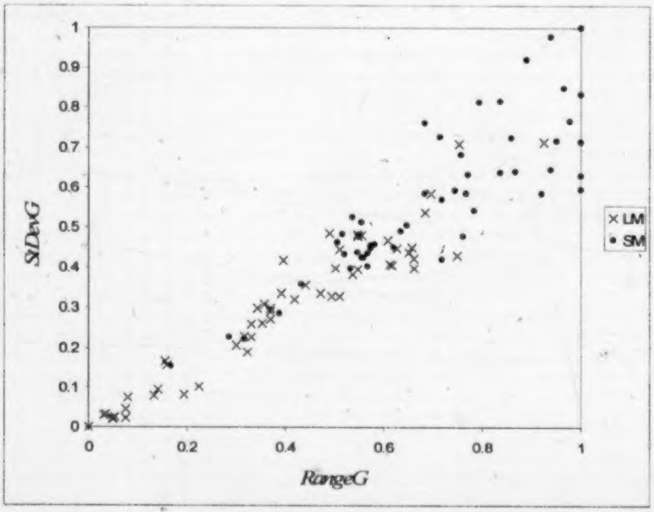


Figure 4-14: Scatter plot of *RangeG* versus *StDevG* for the training set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

To verify the hypothesis that the samples in the test data set were representatives of the same general population from which the training data set was draw, the same statistics and scatter plots computed for the training set were computed for the test set. Table 4-2 depicts statistical information of the features contained in the test data set. From Table 4-2, one can notice that all the explanatory variables had a significant correlation to the class of registration solutions (Large Misregistration and Small Misregistrations). The *MeanG* and *MedianG* had similar mean value (0.503 for *MeanG* and 0.488 for *MedianG*) and standard deviation (0.269 for *MeanG* and 0.262 for *MedianG*). The correlation coefficient between *MeanG* and *MedianG* was $r = 0.99$ suggesting that there was a very large linear correlation between the two variables. The coefficient of determination was $r^2 = 0.98$ suggesting that 98% of the total variation in one variable can be expressed by the linear relation between both variables. These findings confirmed that the hypothesis that the *MeanG* and *MedianG* were related was true.

Figure 4-15 through Figure 4-20 provide a graphical representation of the test set variables. Patterns associated with the class of Large Misregistrations were identified with crosses while patterns associated with the class of Small Misregistrations were identified with dots. Figure 4-15 through Figure 4-17 show samples belonging to the same class grouped together. Figure 4-18 through Figure 4-20 show samples belonging to the same class scattered over the graph. These findings were in agreement to the ones obtained for the training set.

Name	Mean	SD	Min	Max	r
FJIG	0.023	0.006	0.010	0.037	0.809*
MeanG	0.503	0.269	0.000	1.000	0.546*
MedianG	0.488	0.262	0.000	1.000	0.536*
RangeG	0.547	0.292	0.000	1.000	0.595*
StDevG	0.532	0.510	0.000	1.000	0.566*

Table 4-2: List of features of the test set. SD: standard deviation. N: Number of vertebral endplates = 37. r: Pearson coefficient of correlation to the class of registration solutions; for N=37, r>0.325 for significance at p<0.05. * indicates significance at p<0.05.

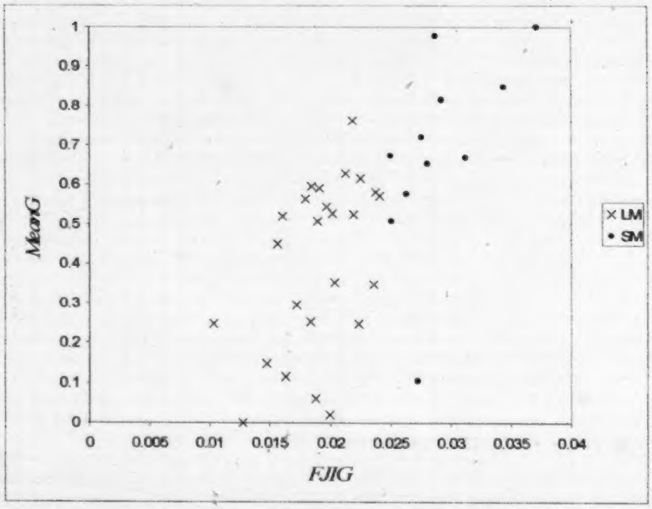


Figure 4-15: Scatter plot of *FJIG* **versus** *MeanG* **for the test set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".**

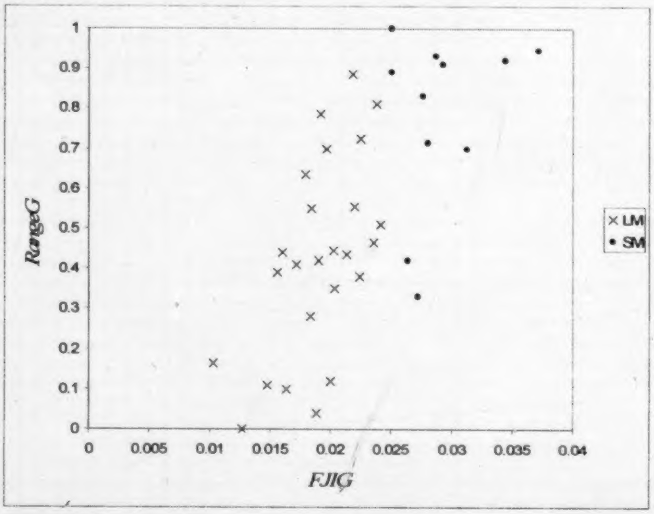


Figure 4-16: Scatter plot of *FJIG* **versus** *RangeG* **for the test set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".**

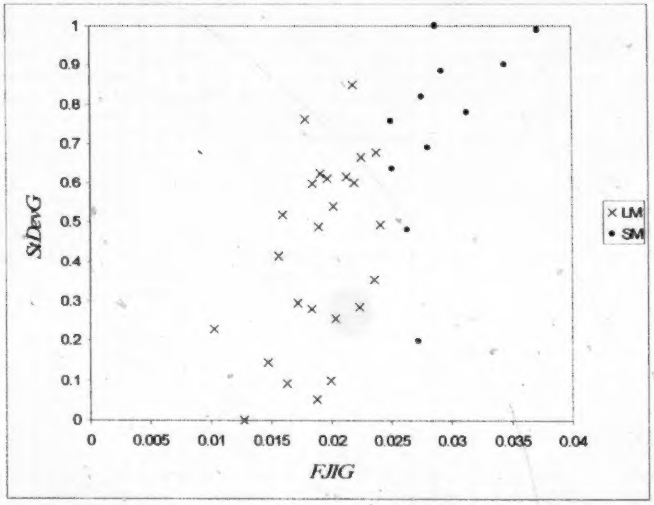


Figure 4-17: Scatter plot of *FJIG* versus *StDevG* for the test set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

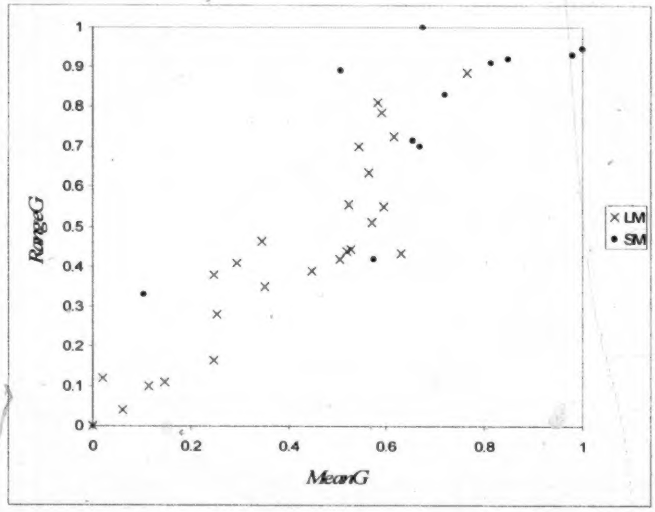


Figure 4-18: Scatter plot of *MeanG* versus *RangeG* for the test set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".

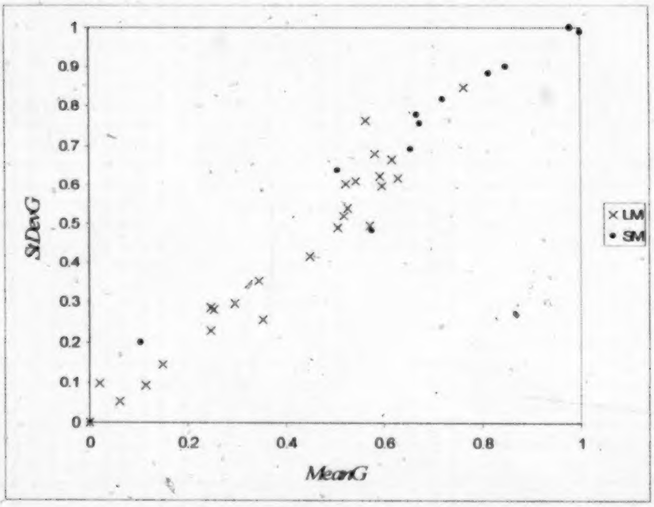


Figure 4-19: Scatter plot of *MeanG* **versus** *StDevG* **for the test set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".**

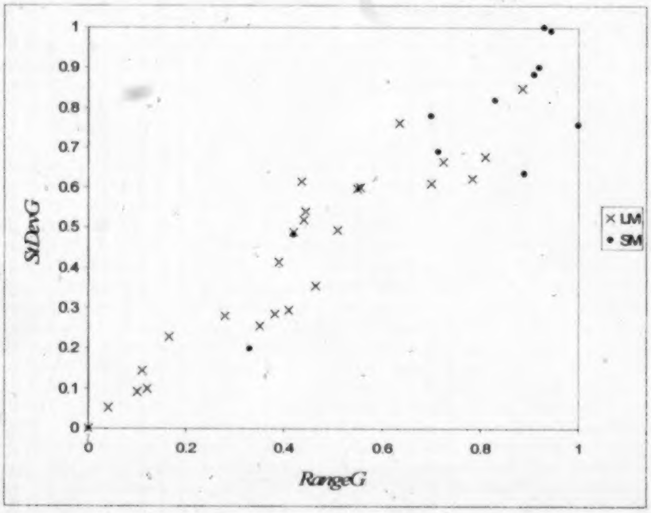


Figure 4-20: Scatter plot of *RangeG* **versus** *StDevG* **for the test set. Patterns labelled as "LM" belong to the class of "Large Misregistrations". Patterns labelled as "SM" belong to the class "Small Misregistrations".**

4.10 Training and testing of the classifiers

To test the hypothesis that the support vector classifier (SVC) with the proposed set of input variables (*FJIG*, *MeanG*, *StDevG*, *RangeG*) could discriminate between good and bad registration solutions, two classification problems were solved.

1. For the first classification problem, the training and testing of the classifiers were done using a 10-fold cross-validation approach. The training data set was divided into 10 non-overlapping groups. Training was done using 9 out of the 10 groups and testing was done in the remaining group. A rotation method was then employed repeating the experiments 10 times. The results obtained after the ten iterations were averaged.

2. For the second classification problem, the training was done using the complete training data set and the testing was done using the testing data set.

The training data for the experiments were linearly scaled to be in the range [-1, 1]. The minimum and maximum values in the training data were then used to scale the testing data for the experiments. To compare the classifiers the McNemar's test was used as suggested in [65] and [17]. This test is based on a Chi-square (χ^2) test for goodness of fit. This test was used to determine if two observed results were different either due to chance or due to a real difference. The χ^2 test was performed following these steps: 1) build a contingency table, as the one shown in Table 4-3; 2) compute the statistics equation (4-16); and 3) if the result of equation (4-16) is greater than $\chi^2_{1,0.95} = 3.8$, then reject the null hypothesis in favour of the hypothesis that the two classifiers have different performance when trained on a particular training set [17].

$C_A \setminus C_B$	NOS Misclassified by C_A	NOS no Misclassified by C_B
NOS Misclassified by C_A	N00	N01
NOS no Misclassified by C_B	N10	N11

Table 4-3: Contingency table used to compare two classifiers: Classifier A (C_A) and Classifier B (C_B). NOS = Number of Samples.

$$\left(\left|N_{01} - N_{10}\right| - 1\right)^2 / \left(N_{01} + N_{10}\right) \qquad (4\text{-}16)$$

The accuracy, sensitivity and specificity of the classifiers were also computed. The accuracy was given by the percentage of registration solutions that were correctly labelled as either Large Misregistration (true positives) or correctly flagged as Small Misregistration (true negatives). The sensitivity (*SN*) and specificity (*SP*) were given by

$$SN = \frac{TP}{TP + FN} \qquad (4\text{-}17)$$

$$SP = \frac{TN}{FP + TN} \qquad (4\text{-}18)$$

with *TP* being the "true positives" (number of registration solutions that were not correctly registered and were classified as having a Large Misregistration); *TN* being the "true negatives" (number of registration solutions that were correctly registered and were classified as having a Small Misregistration); *FP* being the "false positives" (number of registration solutions that were correctly registered and were classified as having a Large Misregistration); *FN* being the "false negatives" (number of solutions that were not correctly registered and were classified as having a Small Misregistration). In clinical practice, a classification approach should have zero (or very close to zero) false negative while the number of false positives is small.

For the SVC, the type of kernel selected was the radial basis function kernel. This kernel can handle non-linear relations between class labels and features and can also perform as a linear kernel for some parameters C (the error penalty term) and γ (the spread for the radial basis kernel) [39]. To select the appropriate values for C and γ, a process of two-level 5-fold cross-validation of the dataset was carried out. In the first level, the sets of parameters used in the cross-validation were: $\gamma \in \{2^{-12}, 2^{-9}, \ldots, 2^{2}\}$ and $C \in \{2^{-2}, 2^{-1}, \ldots, 2^{14}\}$. This level was used to determine the intervals of the parameters that produced the best results. In the second level, based on the intervals obtained in the first level, the sets of parameters used in the cross-validation were: $\gamma \in \{2^{-7.0}, 2^{-6.5}, 2^{-6.0}, 2^{-5.5}, 2^{-5.0}\}$ and $C \in \{2^{6}, 2^{5}, 2^{4}, 2^{3}, 2^{2}, 2^{1}, 2^{0}\}$. The SVC used in the experiments was designed using the LIBSVM [8] (an open source library for support vector machines) implementation in Weka [80] (an open source collection of machine learning algorithms in the Java programming language).

For the decision tree classifier, a C4.5 [54] decision tree implemented in Weka [80] was considered. C4.5 builds decision trees from a set of training data using the concept of Information Entropy. The Information Entropy of a decision tree node is the sum over all classes represented in the node of the proportion of the samples belonging to a particular class times the logarithm in base two of the proportion. The Information Entropy reaches a maximum value of one when only one class is represented at a node and it reaches a minimum value of zero when class sizes at the node are equal. C4.5 uses the fact that each explanatory variable in the data set can be used to make a decision that splits the data set into smaller subsets. C4.5 examines the normalized Information Gain (difference in Information Entropy) that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is the one used to make the decision. The algorithm is then repeated on the resulting smaller sub-trees. The decision tree keeps growing as long as new splits can be found that improve the accuracy on the training set. If left as is, the resulting tree may perform poorly on a test set. To increase the accuracy of the resulting tree, pruning (eliminating splits) needs to be performed. For the C4.5 implementation the level of pruning is controlled by the pruning confidence factor. The pruning confidence factor (*pc*) takes on values between 0 and 1. For *pc* values close to 1, almost no pruning is performed. For *pc* values close to 0, aggressive pruning is performed. To select an appropriate value for *pc* a process of two-level 5-fold cross-validation of the dataset was carried out. In the first level, the sets of parameters used in the cross-validation were: $pc \in \{0.10, 0.20, \ldots, 0.50\}$. This level was used to determine the intervals of the parameters that produced the best results. In the second level, based on the intervals obtained in the first level, the sets of parameters used in the cross-validation were: $pc \in \{0.20, 0.25, 0.30\}$.

For the logistic regression classifier, the logistic regression implementation in Weka [80] was considered. For this implementation there was only one open parameter, the penalty parameter (λ). The penalty parameter controls the amount of shrinkage of the regression coefficients while training the logistic regression classifier [42]. When $\lambda \to \infty$, all the regression coefficients tend to zero. When $\lambda = 0$, the regression coefficients are computed by solving a maximum likelihood estimation [42]. To select an appropriate value for the penalty parameter (λ) a process of two-level 5-fold cross-validation of the dataset was carried out. In the first level, the sets of parameters used in the cross-validation were: $\lambda \in \{2^{-30}, 2^{-29}, \ldots, 2^{-5}\}$. This level was used to determine the intervals of the parameters that produced the best results.

In the second level, based on the intervals obtained in the first level, the sets of parameters used in the cross-validation were: $\gamma \in \{2^{-7.0},\ 2^{-6.5},\ 2^{-6.0},\ 2^{-5.5},\ 2^{-5.0}\}$.

4.11 Results

This section compares the results, in testing, of the Support Vector Classifier (SVC) with those of the Decision Tree (DT) and Logistic Regression (LR) classifiers and evaluates their performance. The results of classifying the cases into "Large Misregistration" and "Small Misregistration" for the first classification problem (using a 10-fold cross-validation approach on the training data set) are summarized in Table 4-4. In terms of accuracy, the DT performed slightly better than the SVC and LR classifiers with an accuracy of 93% compared to 92% for the SVC and 90% for the LR. The SVC and LR detected the highest percentage of cases with Large Misregistration: 89%, compared to 87% of the DT. For the first two-class classification problem, all the classifiers performed slightly better than a radial basis function neural network used in preliminary studies that achieved an average 89% of accuracy in testing.

For the training data set, the SVC and LR classifiers had the largest sensitivity (SN=0.89). The DT had a sensitivity SN=0.87. The largest specificity was achieved by the DT (SP=1.0). It was followed by the SVC (with an SP=0.96) and the LR (with an SP=0.92).

For the first two-class classification problem, Table 4-5 shows the contingency table of SVC vs. LR. Using the results from this table, the McNemar's test score *MT* was equal to 0.5. This result was less than the minimum value ($\chi^2_{1,0.95} = 3.841459$) required to reject the null hypothesis with a 95% of confidence. Therefore, statistically, the SVC and the LR had similar performance. Table 4-6 shows the contingency table of SVC vs. DT. Using the results from this table, the McNemar's test score MT was equal to 0. This result was less than the minimum value ($\chi^2_{1,0.95} = 3.841459$) required to reject the null hypothesis with a 95% of confidence. Therefore, statistically, the SVC and the DT had similar performance.

TO\CO	SVC		DT		LR	
	LM	SM	LM	SM	LM	SM
LM	48(89%)	6(11%)	47(87%)	7(13%)	48(89%)	6(11%)
SM	2(4%)	48(96%)	0(0%)	50(100%)	4(8%)	46(92%)
Accuracy	92%		93%		90%	
SN	0.89		0.87		0.89	
SP	0.96		1.00		0.92	

Table 4-4: First two-class classification problem: Large Misregistrations versus Small Misregistrations. Test results with the SVC, DT, and LR classifiers using a 10-fold crossvalidation approach on the training set. TO: True Output. CO: Classifier Output. LM: Large Misregistrations. SM: Small Misregistrations. SN: Sensitivity. SP Specificity

LR \ SVC	Misclassified by SVC	No Misclassified by SVC
Misclassified by LR	8	2
No Misclassified by LR	0	94
McNemar's Test (MT)Result	$MT = 0.5 < \chi^2_{1,0.95} = 3.841459$	

Table 4-5: First two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus LR using a 10-fold crossvalidation approach on the training data set.

DT \ SVC	Misclassified by SVC	No Misclassified by SVC
Misclassified by DT	6	1
No Misclassified by DT	2	95
McNemar's Test (MT)Result	$MT = 0 < \chi^2_{1,0.95} = 3.841459$	

Table 4-6: First two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus DT using a 10-fold crossvalidation approach on the training data set.

The results of classifying the cases into "Large Misregistration" and "Small Misregistration" for the second classification problem (using the test data set) are summarized in Table 4-7. In terms of accuracy, the SVC outperformed the DT and LR classifiers with accuracy of 86% compared to 68% and 76%, respectively, for the DT and LR. Moreover, the

SVC detected the largest percentage of cases with Large Misregistration: 96%, compared to 64% of the LR and only 52% of the DT.

For the test data set, the SVC had the largest sensitivity (SN=0.96). It was followed by the LR (with an SN=0.64) and the DT (with an SN=0.52). The largest specificity was achieved by both the DT and LR classifiers (SP=1.0). The SVC achieved a specificity of SP=0.67.

For the second two-class classification problem, Table 4-8 shows the contingency table of SVC vs. LR. Using the results from this table, the McNemar's test score *MT* was equal to 0.75. This result was less than the minimum value ($\chi^2_{1,0.95} = 3.841459$) required to reject the null hypothesis with a 95% of confidence. Therefore, statistically, the SVC and the LR had similar performance. Table 4-9 shows the contingency table of SVC vs. DT. Using the results from this table, the McNemar's test score M was equal to 2.4. This result was less than the minimum value ($\chi^2_{1,0.95} = 3.841459$) required to reject the null hypothesis with a 95% of confidence. Therefore, statistically, the SVC and the DT had similar performance.

TO\CO	SVC		DT		LR	
	LM	SM	LM	SM	LM	SM
LM	24(96%)	1(4%)	13(52%)	12(48%)	16(64%)	9(36%)
SM	4(33%)	8(67%)	0(0%)	12(100%)	0(0%)	12(100%)
Accuracy	86%		68%		76%	
SN	0.96		0.52		0.64	
SP	0.67		1.00		1.00	

Table 4-7: Second two-class classification problem: Large Misregistrations versus Small Misregistrations. Test results with the SVC, DT, and LR classifiers on the test set. TO: True Output. CO: Classifier Output. LM: Large Misregistrations. SM: Small Misregistrations. SN: Sensitivity. SP Specificity

LR \ SVC	Misclassified by SVC	No Misclassified by SVC
Misclassified by LR	1	8
No Misclassified by LR	4	24
McNemar's Test (MT)Result	$MT = 0.75 < \chi^2_{1,0.95} = 3.841459$	

Table 4-8: Second two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus LR on the test data set.

DT \ SVC	Misclassified by SVC	No Misclassified by SVC
Misclassified by DT	1	11
No Misclassified by DT	4	21
McNemar's Test (MT)Result	$MT = 2.4 < \chi^2_{1,0.95} = 3.841459$	

Table 4-9: Second two-class problem: Large Misregistration versus Small Misregistration. Contingency table of SVC versus DT on the test data set.

4.12 Discussion

This research was motivated by the observation that there is a need for better evaluation techniques for image registration solutions. With this in mind, a novel methodology for automatically detecting large misregistrations was developed. The main components of the proposed methodology are a novel fuzzy overlap measure and a support vector machines classifier. The results show that, when solving model-to-image registration problems on spine radiographs, the proposed approach achieved between 86% and 92% of classification accuracy for determining whether a registration solution was a Large Misregistration or not.

Results obtained from comparing classifiers with a test set showed that the support vector outperformed a decision tree and logistic regression classifiers. The classification accuracy for the SVC was 86% in testing with a sensitivity SN=0.96 and a specificity of SP=0.67. The classification accuracy for the DT was 68% in testing with a sensitivity SN=0.52 and a specificity of SP=1.0. The classification accuracy for the LR was 76% in testing with a sensitivity SN=0.64 and a specificity of SP=1.0. Even though, the SVC achieved a high sensitivity to Large Misregistrations, work on improving its sensitivity continues because improvements in the sensitivity will translate to early detection of large misregistration guaranteeing a high level of care for the patients. It was observed that 1 record in the test set was misclassified by all the classifiers. This may suggest that the misclassified record was an outlier present in the data set.

The results in Table 4-1 and Table 4-2, show that the fuzzy overlap indicator ($FJIG$) had a statistically significant correlation to the class of type of registration solutions ($r = 0.781$ for the training data set and $r = 0.807$ for the test data set) at $p<0.05$. The image descriptors ($MeanG$, $MedianG$, $RangeG$, and $StDevG$) also showed a statistically significant correlation to the class of the type of registration solutions (r in $[0.550, 0.591]$ for the training data set and r in $[0.536, 0.595]$ for the test data set) at $p<0.05$. It is important to note that the image descriptors were computed only for the pixels in the reference image that corresponded to the pixels in the registered input model that had the maximum membership value.

Figure 4-9 shows a scatter plot of the $FJIG$ versus $MeanG$; Figure 4-10 shows a scater plot of the $FJIG$ versus $RangeG$; and Figure 4-11 shows a scatter plot of the $FJIG$ versus $StDevG$. In all these scatter plots, two distinct clusters (collection of patterns that are close to each other on the screen) can be seen. One cluster is labelled as "LM" (Large Misregistration) and is represented by crosses. The other cluster is labelled as "SM" (Small Misregistration) and is represented by dots. The fact that the data set can be divided by class suggests that the proposed fuzzy overlap measure has a high discriminatory power.

The SVC-based system outperformed systems for the evaluation of registration accuracy based on expert analysis [81], and segmentation of registered images [59] proposed in the literature. In [81], the authors performed a study on the efficacy of visual assessment of registration accuracy on PET to MR images. They considered one axis at the time (only rotation around one axis or translation in an axis at the time). Their results indicated that 100% of the time, experts could detect translational misregistrations of 3 mm or more and rotational misregistrations of 4 degrees or more. They also reported that 60% of the time, the experts could detect misregistrations of more than 2 mm and that 80% of the time, the experts could detect misregistrations of 3 degrees or more. With accuracies over 86%, the SVC-based system is a clear alternative to visual inspection by a domain expert. In [59], a three-phase process (segmentation of the images, computation of principal axes, and determination of a quality measure from contour volumes) was used to distinguish correctly registered solutions from greatly misregistered solutions. Correctly registered solutions were defined as those in which the translation error was less than or equal to 3 mm and the rotation error was less than or equal to 4 degrees following the results presented in [81]. The best result they reported had a sensitivity to large misregistrations of SN=0.6 for a specificity of SP=0.97. The SVC-based approach has better performance considering first that the misregistrations it detected were smaller (above 3 degrees) and second that the SVC-based system achieved a sensitivity of SN=0.96 for a specificity of SP=0.67. It is important to mention that in clinical practice, it is of utmost interest to achieve the largest sensitivity possible to large misregistrations while attaining a high specificity. This is needed to avoid that large misregistration solutions compromise the validity of diagnosis and/or the validity of the planning and evaluating of therapeutic procedures.

Another work that is related to the work presented here is that presented in [15]. In that work, a fuzzy Jaccard index on fuzzy labels was used to drive a registration method and to evaluate the quality of segmentation results. A direct comparison is not possible, because the authors suggest performing segmentation as a means to evaluating registration accuracy. Moreover, because segmentation in itself is a very challenging problem, the SVC-based approach could have a higher applicability in the assessment of the quality of registration solutions.

Although the SVC-based method was designed for estimating the quality of model-to-image registrations of spine radiographs, the proposed approach may be useful in other contexts as well. Since the method does not rely on explicit landmark identification, it should be applicable to the analysis of registration solutions from other imaging modalities and possibly to images of other parts of the body.

All these findings suggest that the proposed system may be a valid option for monitoring the quality of image registration solutions without requiring close supervision from neither a domain expert nor an image registration expert. This could encourage the use of the proposed approach in autonomous image registration applications.

4.13 Summary

This chapter has described a novel approach to use fuzzy overlap measure for the assessment of the quality of image registration solutions. The theory behind such an approach was presented. Two classification problems were used to evaluate the performance of a Support Vector Classifier, using the proposed fuzzy overlap measure, in the categorization of registration solutions. The results suggest that it is possible to differentiate Large Misregistrations from Small Misregistration by using the proposed fuzzy overlap measure, image data, and a support vector classifier. This finding may be useful in identifying when an image registration application requires expert intervention to provide new configuration settings that could improve the registration solutions.

Compared to the decision tree and logistic regression classifiers and compared to the results presented in the literature, the proposed support vector classifier-based system achieved superior performance in terms of classification accuracy. These results may encourage the development of new intelligent systems based on support vector classifiers to automatically assess the quality of registration solutions.

5 Discussion

In this work, a common overlap measure was generalized with the help of fuzzy sets theory to measure overlap either between corresponding anatomical structures of interest in two images or between an image and an anatomical model. To focus on the anatomical structures, the overlap measure was computed on fuzzy set representations of the gradient magnitudes of the images. The applicability of the overlap measure was verified by using it in a similarity measure for image registration and as one of the inputs in a classifier to discriminate acceptable registration solutions from unacceptable ones. Because the clinical focus of this work was on spinal deformities, the main interest was on study of the inclinations of vertebrae. This interest was reflected in the use of the assessment of vertebral endplate rotation as an indicator of registration accuracy.

5.1 Proposed overlap measure in a similarity measure for image registration.

The proposed overlap measure (the Fuzzy Jaccard Index – *FJI*) was applied on fuzzy set representations of gradient magnitudes to focus on finding the best possible match between the gradient magnitudes of the images being registered. However, working only with gradients would discard important information from the intensity values in the images. For this reason, a novel similarity measure, the Combined Fuzzy Jaccard Index (*CFJI*), was considered for image registration. The *CFJI* combined the proposed overlap measure on gradient magnitudes (the *FJI* on gradients or FJI_g) with a version of the proposed overlap measure that computed the overlap on intensity values (the *FJI* on intensities or FJI_I). As a result, the new similarity measure focused on simultaneously aligning corresponding edges and corresponding regions of smooth intensity values. The *CFJI* has two configuration parameters: a weighting factor (α) and a smoothing factor (σ). Both parameters are positive numbers less than or equal to 1. The weighting factor controls the focus of the similarity measure on edges or on intensity values. As the specified value of the weighting factor increases, the focus of the similarity measure on the edges increases and the focus on intensity values decreases. The smoothing factor controls the level of smoothing applied to the image before computing the gradient. As the specified value of the smoothing factor increases, the similarity measure sensitivity to noise decreases along with the similarity measure's ability to deal with weak edges.

To test the hypothesis that the *CFJI* could be a valid alternative as a similarity measure for medical image registration of spine radiographs, a model-to-image registration experiment was set up. Sixteen vertebral endplates were selected and a *CFJI*-based image registration algorithm was used to register a model to the vertebral endplates. The *CFJI* -based registration algorithm was able to precisely register the model to endplates in all the cases (an accuracy of 100%). The correlation coefficient between the expected rotation angle and the computed rotation angle was r=0.99. These results compare favourable with those obtained using a registration algorithm based on the mean squared errors (*MSE*) similarity measure on the same data set. The accuracy of the *MSE*-based registration algorithm was 80%. The correlation coefficient between the expected rotation angle and the computed rotation angle was r=0.76 for the *MSE*-based registration algorithm. These results suggest that the proposed similarity measure works well for images with variable contrast and moderate focus such as the clinical images associated with scoliosis management.

The proposed image registration similarity measure was tested mainly with spine radiographs of patients with scoliosis. Registration of radiographs is challenging due to their variable contrast, moderate focus, low resolution, and variability associated with the representation of three dimensional (3-D) anatomical structures in two dimensional (2-D) images. Because its controlled focus on edges and intensity values information, the proposed similarity measure enhanced the performance of the registration algorithm used in this work in such a way that it could handle those challenges. It is expected that registration algorithms based on the proposed similarity measure will perform well in model-to-image registration of spine images from other modalities (which could be useful in other areas of spine-related research).

These results are important because they suggest that a registration algorithm based on the proposed similarity measure could be used to assess wedge angles in spinal deformities research. As suggested by Aubin et al. [1], the wedge angles obtained from spine radiographs give reliable information that allows partial characterization of the real 3-D vertebral body wedging. However, as Aubin et al. [1] also comment, the location of the maximum wedging angle can not be obtained nor deduced from radiographic measurements. If the location of the maximum wedging angle is required, one can envision using a 3-D model-to-image registration algorithm based on the proposed similarity measure to determine 3-D vertebral wedging.

Regarding the applicability of the proposed similarity measure in single- and multi-modality registration, preliminary results using brain images suggest that it is possible to use the proposed similarity measure in the registration of anatomical structures other than vertebral endplates.

5.2 Proposed overlap measure as a measure of registration quality

Obtaining precise results in the registration of images is important because it allows the extraction of relevant clinical information. Before extracting clinical information from the aligned images, it is necessary to verify the goodness of the alignment. This task is usually done by visual inspection. Unfortunately, this approach is time consuming and highly dependant on the level of expertise of the radiologist, physician, or other clinical expert. This complicates the adoption of image registration methods in clinical practice. To solve this problem, a classification strategy, using the proposed overlap measure, was developed to automatically assess whether the registration was successful or not.

The proposed overlap measure (the Fuzzy Jaccard Index – *FJI*) was applied on fuzzy set representations of gradient magnitudes to measure the fuzzy overlap between a model and the region of interest containing the vertebral endplate under study. The *FJI* on gradient magnitudes (*FJIG*) has one configuration parameter: a smoothing factor (σ). The smoothing factor attains positive values in [0, 1]. The smoothing factor is the standard deviation of the Gaussian smoothing kernel used to filter the image before computing the gradient magnitudes.

To test the hypothesis that the *FJIG* was related to the category of the registration solution quality (large misregistration or Small misregistration), the correlation coefficient between the *FJIG* and the class of registration solutions was computed. The correlation coefficient was equal to r=0.78 which is equivalent to an $r^2 = 0.61$. The value of r^2 indicates that 61%of the total variation in the class of registration solutions can be explained by the linear relation between *FJIG* and the class of registration solution. This result suggests that the *FJIG* can, indeed, help in discriminating large misregistrations from Small misregistrations.

The main thrust of this work was not only to develop an overlap measure that was related to registration quality but also to demonstrate that taken together with a group of image descriptors, the resulting indices could estimate registration quality accurately. The contribution of individual indices to this estimation should be tested more thoroughly and formally with growth of the data set over time.

To test the hypothesis that a Support Vector Classifier (SVC), using the *FJIG*, could predict the quality of registration solutions sufficiently well to be used in an automated system for detecting large misregistrations, a model-to-image registration experiment was set up. 141 vertebral endplates were selected. 104 vertebral endplates were used as a training set and 37 vertebral endplates were used as a test set. A *CFJI*-based image registration algorithm was used to register a model to the vertebral endplates using a fix set of values for the weighting and smoothing factors ($\alpha = 0.5$ and $\sigma = 2.0$ respectively). Fifty four of the registration solutions in the training set were labelled as large misregistrations and fifty of the registration solutions were labelled as Small misregistration solutions. The overlap measure and some image descriptors were computed for each vertebral endplate. The resulting data set was used to train a support vector classifier, a decision tree classifier, and a logistic regression classifier. The performance of the classifiers was assessed using a 10-fold cross-validation approach. The support vector classifier achieved an accuracy of 92%. This result was similar to the one obtained using a decision tree classifier (93%) and a logistic regression classifier (90%).

A *CFJI*-based image registration algorithm was used to register a model to the 37 vertebral endplates in the test set using a set of fixed values for the weighting and smoothing factors ($\alpha = 0.5$ and $\sigma = 2.0$ respectively). Twenty five of the registration solutions were labelled as large misregistrations and twelve of the registration solutions were labelled as

Small misregistration solutions. The overlap measure and some image descriptors were computed for each vertebral endplate. The resulting data set was used to assess the performance of a SVC in discriminating the quality of registration solutions. The performance was assessed using the complete training set for configuring the classifiers and using the test set to evaluate the performance. The support vector classifier achieved an accuracy of 86%. This result was better than the one obtained using a logistic regression classifier (76%) and a decision tree classifier (68%). The high accuracy obtained by the support vector classifier suggests that the proposed system is a viable alternative for automatically evaluating medical image registration solutions.

The classification results were used to determine whether or not the location and orientation of the endplates under study could be used to calculate some clinical indicators of scoliosis severity with the goal of using the proposed system for automatically assessing scoliosis severity in clinical practice. The results were very promising showing a difference of less than 3° with respect to the measurements made by a clinical expert. This difference is less than the accepted intra- inter-observer error in measuring scoliosis severity [46].

Because no assumption was made regarding the image modalities under study, it is expected that the proposed system will perform well in evaluating the registration results of other model-to-image registration applications provided that the boundaries of the structures of interest are identifiable and an adequate model is used to match the anatomical structure of interest.

5.3 Limitations

This document describes the development of fuzzy overlap measures and their use for controlling registration algorithms and for assessing registration quality in a model-to-image registration application.

Some fundamental assumptions and limitations of this work should be considered when analysing its results. The limitations of this work are related to the size and nature of the data set, the available technology, and the clinical protocol.

Although radiographs are widely used for scoliosis management and spinal deformities research, it is important to note that they show an incomplete 2-D description of a 3-D scoliotic deformity. This is why there is no single model that would work adequately for all the possible ways in which a vertebral endplate would look like in a radiograph.

There were limitations related to the small sample size. We had 141 regions of interest (with their corresponding vertebral endplates) for 18 patients. The regions of interest were divided into two groups: a training set with 104 regions of interest and a test set with 37 regions of interest. The data set may not be representative of the general population of scoliosis images. This leads to concerns about overfitting and potential bias on the part of the classifiers. These concerns were alleviated after evaluating the support vector classifier on the test data set and having similar results than those obtained with the training data set using a ten-fold cross-validation approach.

All the reported simulations were performed with two dimensional (2-D) images. Because no assumption about image dimensionality was done in the development of the proposed framework, it is expected that the results would be extensible to work with three dimensional (3-D) images.

Finally, the results are dependant on the image modalities used, the resolution of the images, and the distribution of the misregistration errors created. To address this issue, simulations involving a larger dataset should be carried out.

6 Conclusion and recommendations for future work

6.1 Conclusions

The engineering intent of this work was to develop a novel fuzzy sets overlap measure that could be used to assess the quality of image registration results. This goal was divided into two main objectives. The first objective was to study the proposed overlap measure to control image registration processes. The second objective was to use the overlap measure to assess the quality of registration results in a model-to-image registration application.

Regarding the first objective, this work has assessed the applicability of the proposed overlap measure as a similarity measure for a series of image registration tasks that included single-modality registration of brain images, multi-modality registration of brain images, and model-to-image registration of sixteen vertebral endplates images. Compared to the registration algorithm using the Mean Squared Error similarity measure, the registration algorithm using the proposed similarity measure achieved superior performance in all the cases. Moreover, in solving the model-to-image registration problems, the registration algorithm using the proposed similarity measure could precisely register the vertebral endplate in all the cases while the registration algorithm using the mean squared errors similarity measure could register precisely 80% of the vertebral endplates. The good results obtained with the proposed similarity measure may be related to the fact that while most similarity measures treat the same all pixels, the proposed similarity measure gives special attention to edges. Because of this, the proposed similarity measure exploits the spatial information contained in the location of edges. By giving special attention to the edges while using the information contained in the intensity values, the proposed similarity measure mimics the way in which human beings analyze images (i.e., grouping similar pixels to make sense of the images). This fact makes it more robust than other similarities in dealing with medical images.

Regarding the second objective, this work has assessed the applicability of a support vector classifier using the proposed overlap measure for assessing the quality of registration results in model-to-image registration tasks. Compared to logistic regression and decision tree classifiers the support vector classifier achieved superior classification accuracy. The support vector classifier had similar performance to that of the other classifiers when tested using a 10-fold cross-validation approach on the training data. The support vector classifier, however, clearly outperformed the other classifiers when tested on a previously unseen test set. The good results obtained with the support vector classifier may be related to the fact that the support vector machines generally produce smooth decision surfaces that maximize the margin between classes. This fact makes the support vector classifier more robust than the other classifiers in dealing with imprecise class labels.

The clinical intent of this work was to develop a system for measuring inclinations of vertebrae in spine radiographs of patients with scoliosis. This goal was achieved with an image registration algorithm based on the proposed similarity measure. When properly configured, the registration algorithm produced registration results with a very high correlation (r=0.99) to the expected angulations. When the registration algorithm was not properly configured, a support vector classifier (using the proposed overlap measure as one of its inputs) would indicate that there was a misregistration. In this way, misregistrations results would not be used in computing the inclinations. This is needed to avoid having large misregistration results compromise the validity of the measurements.

In conclusion, it is possible to differentiate Large Misregistrations from Small Misregistration by using the proposed fuzzy overlap measure, image data, and a support vector classifier. This finding may be useful in identifying when an image registration application requires domain expert intervention to provide new configuration settings that could improve the registration solutions.

6.2 Recommendations for future work

The designed system was shown to be able to detect reliably large misregistrations in model-to-image registrations applications associated with scoliosis management and research. Further development of this system could focus on:

1. Developing a statistical framework for the analysis of overlap computed over different subsets of images in such a way that new application areas can be explored for the proposed system.

2. Designing a set of classification problems in which the registration solution is assigned to one of four categories: excellent, good, fair, or bad. The result of this classification could be used in a system that would adjust the registration parameters automatically depending on the results of the classification.

3. Using the proposed overlap measure as the basis for developing neighbourhood-based overlap measures. This recommendation is based on the observation by Van der Weken et al. [16] that when using neighbourhood-based similarity measures to evaluate image quality, the results coincided better with human perception in comparison with the pixel-based similarity measures.

4. Providing the registration algorithms with good initialization parameters to reduce the time required to perform the model-to-image registration that currently stands at about 1 minute per vertebral endplate on a 3GHz Intel Pentium machine with 2GB of memory. One way of finding a good set of initial rotation parameters might be as follows:

    i. Identifying the centre of the top and bottom vertebral endplates of the curve under study and the centre of the vertebra at the apex of the scoliosis curve under study.
    ii. Computing an interpolating spline going through the points identified in the previous step.
    iii. Finding the spline slopes at the locations of the candidate endplates.
    iv. Using angles that are orthogonal to the slopes of the spline. These are good initial rotation parameters because if the vertebrae are not wedged then the vertebral endplates are orthogonal to the scoliotic curve.

References

[1] C.E. Aubin, J. Dansereau, Y. Petit, F. Parent, J.A. de Guise, and H. Labelle, "Three-dimensional measurement of wedged scoliotic vertebrae and intervertebral disks," *European Spine Journal*, vol. 7, no. 3, pp. 59–65, 1998.

[2] M.J. Beuerlein, V.J. Raso, D.L. Hill, M.J. Moreau, and J.K. Mahood, "Changes in alignment of the scoliotic spine in response to lateral bending," *Spine*, vol. 28, no. 7, pp. 693–698, 2003.

[3] C.M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Claredon, 1995.

[4] I. Bloch, "Fuzzy sets in image processing," *Proceedings of the 1994 ACM Symposium on Applied Computing*, pp. 175-179, Phoenix, 1994.

[5] I. Bloch, "Fuzzy spatial relationships for image processing and interpretation: a review," *Image and Vision Computing*, vol. 23, pp. 89–110, 2005.

[6] I. Bloch and A. Ralescu, "Directional relative position between objects in image processing: a comparison between fuzzy approaches," *Pattern Recognition*, vol. 36, pp. 1563–1582, 2003.

[7] W. Burgoyne and J. Fairbank, "The management of scoliosis," *Current Orthopaedics*, vol. 11, no. 5, pp. 323-331, 2001.

[8] C.-C. Chang and C.-J. Lin, (software) LIBSVM: A Library for Support Vector Machines [online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm/, 2001.

[9] S.M. Chen, "Measures of similarity between vague sets," *Fuzzy Sets and Systems*, vol. 74, pp. 217–223, 1995.

[10] S.M. Chen, M.S. Yeh, and P.Y. Hsiao, "A comparison of similarity measures of fuzzy values," *Fuzzy Sets and Systems*, vol. 72, pp. 79–89, 1995.

[11] J. Cheung, D.J. Wever, A.G. Veldhuizen, J.P. Klein, B. Verdonck, R. Nijlunsing, J.C. Cool, and J.R. van Horn, "The reliability of quantitative analysis on digital images of the scoliotic spine," *European Spine Journal*, vol. 11, no. 6, pp. 535-542, 2002.

[12] K. Cios, W. Pedrycz, and R. Swiniarski, *Data Mining Methods for Knowledge Discovery*. Boston: Kluwer Academic Publishers , 1998.

[13] J.R. Cobb, "Outline for the study of scoliosis, instructional course lectures," *The American Academy of Orthopedic Surgeons*, vol. 5, pp. 261-275, 1948.

[14] W.G. Cochran, "Problems arising in the analysis of a series of similar experiments," *Supplement to the Journal of the Royal Statistical Society*, vol. 4, no. 1, pp. 102-118, 1937.

[15] W.R. Crum, O. Camara, and D.L.G. Hill, "Generalized overlap measures for evaluation and validation in medical image analysis," *IEEE Transactions on Medical Imaging*, vol. 25, no. 11, pp. 1451-1461, 2006.

[16] D.V. der Weken, M. Nachtegael, and E.E. Kerre, "Using similarity measures and homogeneity for the comparison of images," *Image and Vision Computing*, vol. 22, no. 9, pp. 695-702, 2004.

[17] T.G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, pp. 1895-1923, 1998.

[18] J. Fan and W. Xie, "Some notes on similarity measure and proximity measure," *Fuzzy Sets and Systems*, vol. 101, pp. 403–412, 1999.

[19] J.M. Fitzpatrick and J. West, "The distribution of target registration error in rigid-body point-based registration," *IEEE Transactions on Medical Imaging*, vol. 20, no. 9, pp. 917–927, 2001.

[20] J.M. Fitzpatrick, D.L.G. Hill, Y. Shyr, J. West, C. Studholme, and C.R. Maurer, Jr., "Visual assessment of the accuracy of retrospective registration of MR and CT images of the brain," *IEEE Transactions on Medical Imag*ing, vol. 17, no. 4, pp. 571–585, 1998.

[21] C. Goldberg, M. Kaliszer, D.P. Moore, E.E. Fogarty, and F.E. Dowling, "Surface topography, Cobb angles, and cosmetic change in scoliosis," *Spine* vol. 26, no. 4, pp. E55-E63, 2001.

[22] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Don Mills, Ontario: Addison-Wesley, 1989.

[23] M. Gstoettner, K. Sekyra, N. Walochnik, P. Winter, R. Wachter, and C.M. Bach, "Inter- and intraobserver reliability assessment of the Cobb angle: manual versus digital measurement tools," *European Spine Journal*, vol. 16, no. 10, pp.1587–1592, 2007.

[24] J.V. Hajnal, D.L.G. Hill, and D.J. Hawkes, *Medical Image Registration*. Boca Raton: CRC Press, 2001.

[25] A. Hamadeh and P. Cinquin, "Kinematic study of lumbar spine using functional radiographies and 3D / 2D registration," in *Proc. CVRMed-MRCAS'97 (LNCS 1205)*, 1997, pp. 109-118

[26] V.M. Haughton, B. Rogers, M.E. Meyerand, and D.K. Resnick, “Measuring the axial rotation of lumbar vertebrae in vivo with MR imaging,” *American Journal of Neuroradiology*, vol. 23, no. 7, pp. 1110-1116, 2002

[27] D.L.G. Hill, P.G. Batchelor, M. Holden, and D.J. Hawkes, “Medical image registration,” *Physics in Medicine and Biology*, vol. 46, no. 3, pp. R1-R45, 2001.

[28] D.L. Hill, D.C. Berg, V.J. Raso, E. Lou, N.G. Durdle, J.K. Mahood, and M.J. Moreau, “Evaluation of a laser scanner for surface topography,” in *Research into spinal deformities 3*, A.H. Tanguy and B. Peuchot, Eds., Oxford: IOS Press, 2002, pp. 90-95.

[29] K. Hirota and W. Pedrycz, “Fuzzy Computing for Data Mining,” *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1575-1600, 1999.

[30] M. Holden, D.L.G. Hill, E.R.E. Denton, J.M. Jarosz, T.C.S. Cox, T. Rohlfing, J. Goodey, and D.J. Hawkes, “Pixel similarity measures for 3-D serial MR brain image registration,” *IEEE Transactions on Medical Imaging*, vol. 19, no. 2, pp. 94–102, 2000.

[31] D.W. Hosmer and S. Lemeshow. *Applied Logistic Regression*, 2nd Ed. New York: Wiley, 2000.

[32] R.H. Huesman, G.J. Klein, J.A. Kimdon, C. Kuo, and S. Majdumdar, “Deformable registration of multimodal data including rigid structures,” *IEEE Transactions on Nuclear Science*, vol. 50, no. 3, pp. 389-392, 2003.

[33] L. Ibáñez and W. Schroeder, *The ITK Sofware Guide - ITK 1.4*. New York: Kitware, Inc., 2003.

[34] B. Jahne, *Practical Handbook on Image Processing for Scientific and Technical Applications*. New York: CRC Press, 2004.

[35] A.K. Jain, R.P.W. Duin, and J. Mao, “Statistical pattern recognition: A review,” *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, vol. 22, no. 1, pp 4-27, 2000.

[36] J.L. Jaremko, P. Poncet, J. Ronsky, J. Harder, J. Dansereau, H. Labelle, and R.F. Zernicke, “Estimation of spinal deformity in scoliosis from torso surface cross sections,” *Spine*, vol. 26, no. 14, pp. 1583-1591, 2001.

[37] J.L. Jaremko, P. Poncet, J. Ronsky, J. Harder, J. Dansereau, H. Labelle, and R.F. Zernicke, “Genetic algorithm-neural network estimation of Cobb angle from torso asymmetry in scoliosis,” *Journal of Biomechanical Engineering-Transactions of ASME*, vol. 124, pp. 496-503, 2002.

[38] J.R. Jiménez-Alaniz, V. Medina-Bañuelos, and O. Yáñez-Suárez, "Data-driven brain MRI segmentation supported on edge confidence and a priori tissue information," *IEEE Transactions on Medical Imaging*, vol. 25, no. 1, pp. 74-83, 2006.

[39] S.S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Computation*, vol. 15 no. 7, pp. 1667–1689, 2003.

[40] N. Khouri, R. Vialle, P. Mary, and C. Marty, "Idiopathic scoliosis. Strategy, pathophysiology, and deformity analysis," *EMC. Rhumatologie Orthopédie*, vol. 1, pp. 17-44, 2004.

[41] H.S. Kim, S. Ishikawa, Y. Ohtsuka, H. Shimizu, T. Shinomiya, and M.A. Viergever, "Automatic scoliosis detection based on local centroids evaluation on Moiré topographic images of human backs," *IEEE Transactions on Medical Imaging*, vol. 20, no. 12, pp. 1314-1320, 2001.

[42] S. Le Cessie and J.C. van Houwelingen, "Ridge estimators in logistic regression," *Applied Statistics*, vol. 41, no. 1, pp. 191-201, 1992.

[43] X.C. Liu, J.G. Thometz, R.M. Lyon, and J. Klein, "Functional classification of patients with idiopathic scoliosis assessed by the quantec system," *Spine*, vol. 26, no. 11, pp. 1274-1279, 2001.

[44] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens, "Multimodality image registration by maximization of mutual information," *IEEE Transactions on Medical Imaging*, vol. 16, no. 2, pp. 187-198, 1997.

[45] J.B.A. Maintz and M.A. Viergever, "A survey of medical image registration," *Medical Image Analysis*, vol. 2, no, 1, pp. 1-36, 1998.

[46] R.T. Morrissy, G.S. Goldsmith, E.C. Hall, D. Kehl, and G.H.Cowie, "Measurement of the Cobb angle on radiographs of patients who have scoliosis. Evaluation of intrinsic error," *Journal of Bone and Joint Surgery – American Volume*, vol. 72, no. 3, pp. 320-327, 1990.

[47] K.-R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181-201, 2001.

[48] I.P.I. Pappas, M. Styner, P. Malik, L. Remonda, and M. Caversaccio, "Automatic method to assess local CT–MR imaging registration accuracy on images of the head," *American Journal of Neuroradiology*, vol. 26, pp. 137-144, 2005.

[49] C.P. Pappis and N.I. Karacapilidis, "A comparative assessment of measures of similarity of fuzzy values," *Fuzzy Sets and Systems*, vol. 56, pp. 171–174, 1993.

[50] W. Pedrycz and F. Gomide, *An Introduction to Fuzzy Sets: Analysis and Design*. Cambridge: MIT Press, 1998.

[51] G.P. Penney, J. Weese, J.A. Little, P. Desmedt, D.L.G. Hill, and D.J. Hawkes, "A comparison of similarity measures for use in 2-D-3-D medical image registration," *IEEE Transactions on Medical Imag*ing, vol. 17, no. 4, pp. 586-595, 1998.

[52] J.P.W. Pluim, J.B.A. Maintz, and M.A. Viergever, "Image registration by maximization of combined mutual information and gradient information," *IEEE Transactions on Medical Imaging*, vol. 19, no. 8, pp. 809-814, 2000.

[53] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in C—The Art of Scientific Computing*. Cambridge: Cambridge University Press, 1992

[54] J.R. Quinlan, *C4.5 Programs for Machine Learning*, San Francisco: Morgan Kaufmann, 1993.

[55] L. Ramirez, N.G. Durdle, and V.J. Raso, "Medical image registration in computational intelligence framework: A review," in *Proceedings of the Canadian Conferences on Electrical and Computer Engineering*, vol. 2, pp. 1021-1024, Montreal, 2003.

[56] L. Ramirez, N.G. Durdle, V.J. Raso, and D.L. Hill, "A support vector machines classifier to assessing the severity of idiopathic scoliosis from surface topography," *IEEE Transactions on Information Technology in Biomedicine*, vol. 10, no. 1, pp. 84-91, 2006.

[57] L. Ramirez, N.G. Durdle, and V.J. Raso, "A parameters selection scheme for medical image registration," in *Proceedings of the Meeting of the North American Fuzzy Information Processing Society (NAFIPS) 2006*, pp. 505-510, Montreal, 2006.

[58] J.W. Roach, "Adolescent idiopathic scoliosis," *Orthopedic Clinics of North America*, vol. 30, pp. 353-365, 1999.

[59] C.E. Rodriguez-Carranza and M.H. Loew, "Design and evaluation of an automatic procedure for detection of large misregistration of medical images," *IEEE Transactions on Medical Imaging*, vol. 22, no. 11, pp. 1445-1457, 2003.

[60] B.P. Rogers, V.M. Haughton, K. Arfanakis, and E. Meyerand, "Application of image registration to measurement of intervertebral rotation in the lumbar spine, *Magnetic Resonance in Medicine*, vol. 48, no. 6, pp. 1072-1075, 2002.

[61] A. Rosenfeld, "The fuzzy geometry of image subsets," *Pattern Recognition Letters*, vol. 2, pp. 311–317, 1984.

[62] T.J. Ross, *Fuzzy Logic – with Engineering Applications 2nd ed*. San Francisco, CA: John Wiley & Sons, 2004.

[63] E.H. Ruspini, P.P. Bonissone, and W. Pedricz, *Handbook of Fuzzy Computation*. Philadelphia, USA: IOP Publishing, 1998.

[64] K.M. Simonson, S.M. Drescher Jr., and F.R. Tanner, "A statistics-based approach to binary image registration with uncertainty analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 112-125, 2007.

[65] A.E. Smith, C. D. Nugent, and S. I. McClean, "Evaluation of inherent performance of intelligent medical decision support systems: Utilizing neural networks as an example," Artificial Intelligence in Medicine, vol. 27, no. 1, pp. 1–27, 2003.

[66] I.A.F. Stokes and D.D. Aronsson, "Disc and vertebral wedging in patients with progressive scoliosis," *Journal of Spinal Disorders*, vol. 14, no. 4, pp. 317–322, 2001.

[67] I.A.F. Stokes and M.S. Moreland, "Concordance of back surface asymmetry and spine shape in idiopathic scoliosis," *Spine*, vol. 14, no. 1, pp. 73-78, 1989.

[68] C. Studholme, DL.L.G. Hill, and D.J. Hawkes, "An overlap invariant entropy measure of 3-D medical image alignment," *Pattern Recognition*, vol. 32, pp. 71-86, 1998.

[69] T.N. Theologis, J.C. T. Fairbank, A.R. Turner-Smith and T. Pantazopoulos, "Early detection of progression in adolescent idiopathic scoliosis by measurement of changes in back shape with the integrated shape imaging system scanner," *Spine*, vol. 22, no. 11, pp. 1223-1227, 1997.

[70] M. Thomsen and R. Abel, "Imaging in scoliosis from the orthopaedic surgeon's point of view," *European Journal of Radiology*, vol. 58, pp. 41–47, 2006.

[71] D. Tomazevic, B. Likar, T. Slivnik, and F. Pernus, "3-D/2-D registration of CT and MR to X-ray images," *IEEE Transactions on Medical Imag*ing, vol. 22, no. 11, pp. 1407-1416, 2003.

[72] A.R. Turner-Smith, J.D. Harris, G.R. Houghton, and R.J. Jefferson, "A method for analysis of back shape in scoliosis," *Journal of Biomechanics*, vol. 21, no. 6, pp. 497-509, 1988.

[73] M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22-38, 1999

[74] R. Valdés-Cristerna, V. Medina-Bañuelos, and O. Yáñez-Suárez, "Coupling of radial-basis network and active contour model for multispectral brain MRI segmentation," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 3, pp. 459-470, 2004.

[75] J. Van Cleynenbreugel, F. Schutyser, J. Goffin, K. Van Brussel, and P. Suetens, "Image-based planning and validation of C1–C2 transarticular screw fixation using personalized drill guides," *Computer Aided Surgery*, vol. 7, no. 1, pp. 41-48, 2002.

[76] V.N. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988-999, 1999.

[77] V.N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[78] W.J. Wang, "New similarity measure on fuzzy sets and on elements," *Fuzzy Sets and Systems*, vol. 85, pp. 305–309, 1997.

[79] W.M. Wells, III, P. Viola, H. Atsumi, S. Nakajima, and R. Kikinis, "Multimodal volume registration by maximization of mutual information," *Medical Image Analysis*, vol. 1, no. 1, pp. 35-51, 1996.

[80] I.H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques* 2nd Edition, San Francisco: Morgan Kaufmann, 2005.

[81] J.C.H. Wong, C. Studholme, D.J. Hawkes, and M.N. Maisey, "Evaluation of the limits of visual detection of image misregistration in a brain fluorine-18 fluorodeoxyglucose PET-MRI study," *European Journal of Nuclear Medicine*, vol. 24, no. 6, pp. 642-650, 1997.

[82] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338-352, 1965.

[83] L. Zadeh, "Outline of a new approach to the analysis of complex systems and decision process," *IEEE Transactions on Systems Man and Cybernetics*, vol. 3, pp. 28-44, 1973.

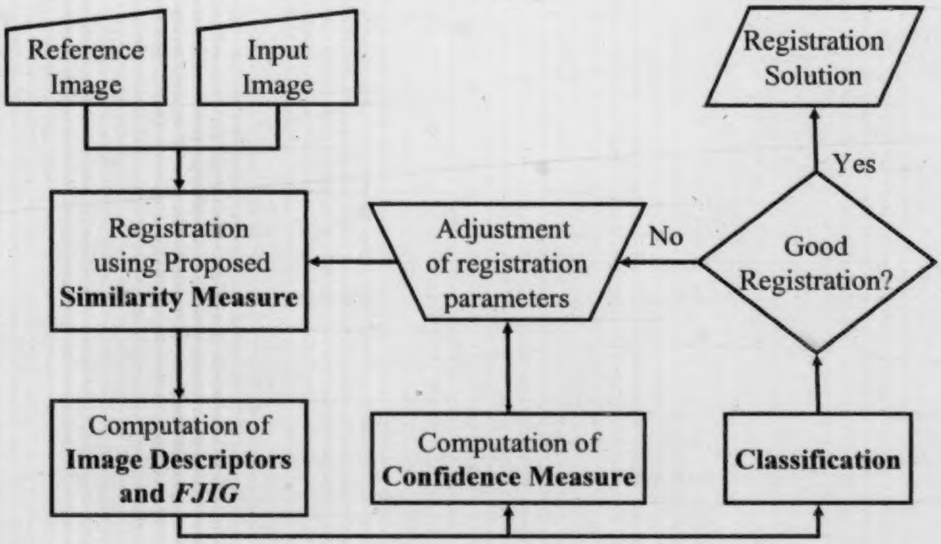
[84] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977-1000, 2003.

APPENDICES

Appendix 1: Use of the System

Here is how the use of the proposed framework works:

- An input image is registered to a reference image. The registration method uses the proposed similarity measure.
- Once registration is completed, image descriptors and the confidence measure are computed.
- The classifier strategy is used to discriminate good registration solutions from bad registration solutions.
- If the registration is good, the registration solutions can be used as is.
- If the registration is not good, either an expert or an automated control system should adjust the registration parameters and resume the registration process. This is important because by using the information conveyed on the proposed confidence measure, it is possible to develop automated registration systems.


Reference Image
Input Image
Registration Solution
Yes
Registration using Proposed **Similarity Measure**
Adjustment of registration parameters
No
Good Registration?
Computation of **Image Descriptors and *FJIG***
Computation of **Confidence Measure**
Classification

Appendix 2: Proposed Similarity Measure

Algorithm for computing the proposed similarity measure

Given	Two overlapping images A and B
Defined	α, the weighting factor σ, the smoothing factor
Initialization	Calculate the gradient magnitudes for images A and B. ($\nabla_\sigma(A)$ and $\nabla_\sigma(B)$)
Processing	For all the pixels in the region of overlap $j \in \{1,2,\ldots,N\}$ a. Calculate the FJI_g by: $FJI_g(A,B) = \frac{\sum_{j=1}^{N} min(\nabla_\sigma(A(x_j)), \nabla_\sigma(B(x_j)))}{\sum_{j=1}^{N} max(\nabla_\sigma(A(x_j)), \nabla_\sigma(B(x_j)))}$ b. Calculate the FJI_i by: $FJI_i(A,B) = \frac{\sum_{j=1}^{N} min(A(x_j), B(x_j))}{\sum_{j=1}^{N} max(A(x_j), B(x_j))}$ Determine the Combined Fuzzy Jaccard Index by: $CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_i(A,B)$
Result	The Combined Fuzzy Jaccard Index $CFJI$

Source code for computing the proposed similarity measure

```
/*=========================================================================

  Program:   Insight Segmentation & Registration Toolkit
  Module:    $RCSfile: itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric.h,v $
  Language:  C++
  Date:      $Date: 2007/08/09 20:04 $
  Version:   $Revision: 0.01 $

  Copyright (c) Lino Ramirez. All rights reserved.

     This software is distributed WITHOUT ANY WARRANTY; without even
     the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
     PURPOSE.  See the above copyright notices for more information.

=========================================================================*/
#ifndef __itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric_h
#define __itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric_h

#include "itkImageToImageMetric.h"

#include "itkGradientMagnitudeRecursiveGaussianImageFilter.h"
#include "itkRescaleIntensityImageFilter.h"
#include "itkNeighborhoodOperatorImageFilter.h"
#include "itkPoint.h"
#include "itkCastImageFilter.h"
#include "itkResampleImageFilter.h"

#include "itkCovariantVector.h"

namespace itk
{
 /** \class CombinedJaccardIndexIntensityAndGradientImageToImageMetric
    \brief Computes a combined Jaccard Index from gradient and Intensity
    information between two images to be registered.

    The type of Intensity Information implemented in this class
    is given by the equation JII

    \f[  \frac{ min(I(A),I(B) } { max(I(A), I(B)) }  \f]

    Where \$ I(A) \$ is the Intensity of image \$ A \$, rescaled to be in [0,1]
         \$ I(B) \$ is the Intensity of image \$ B \$, rescaled to be in [0,1]


    the type of gradient information implemented in this class is given by
    the equation JIG
    \f[  \frac{ min(G(A),G(B) } { max(G(A), G(B)) }  \f]

    Where \$ G(A) \$ is the gradient of image \$ A \$, rescaled to be in [0,1]
         \$ G(B) \$ is the gradient of image \$ B \$, rescaled to be in [0,1]

    The resulting measure is equal to

    \f[ \alpha * JIG(A,B) + ( 1 - \alpha ) * JII(A,B) \f] .

    Where \$ JII \$ is the Jaccard Index Intensity of images \$ A \$ and \$ B\$.
         \$ JIG \$ is the Jaccard Index Gradient of images \$ A \$ and \$ B\$.
         \$ \alpha \$ is a factor in [0, 1] that controls the impact of the
                   JII and JIG in the resulting measure.

    This class is templated over the type of the fixed and moving
    images to be compared.

 \ingroup RegistrationMetrics */
```

```
template <class TFixedImage, class TMovingImage>
class ITK_EXPORT CombinedJaccardIndexIntensityAndGradientImageToImageMetric :
public ImageToImageMetric<TFixedImage, TMovingImage>
{
public:
  /** Standard class typedefs. */
  typedef CombinedJaccardIndexIntensityAndGradientImageToImageMetric Self;
  typedef ImageToImageMetric<TFixedImage, TMovingImage> Superclass;
  typedef SmartPointer<Self>                     Pointer;
  typedef SmartPointer<const Self>                  ConstPointer;

  /** Method for creation through the object factory. */
  itkNewMacro(Self);


  /** Run-time type information (and related methods). */
  itkTypeMacro(CombinedJaccardIndexIntensityAndGradientImageToImageMetric, ImageToImageMetric);

  /** Types transferred from the base class */
  typedef typename Superclass::RealType                RealType;
  typedef typename Superclass::TransformType           TransformType;
  typedef typename Superclass::TransformPointer        TransformPointer;
  typedef typename Superclass::TransformParametersType
   TransformParametersType;
  typedef typename Superclass::TransformJacobianType
   TransformJacobianType;
  typedef typename Superclass::GradientPixelType       GradientPixelType;
  typedef typename Superclass::InputPointType          InputPointType;
  typedef typename Superclass::OutputPointType         OutputPointType;
  typedef typename Superclass::MeasureType             MeasureType;
  typedef typename Superclass::DerivativeType          DerivativeType;
  typedef typename Superclass::FixedImageType          FixedImageType;
  typedef typename Superclass::FixedImageType::PixelType  FixedImagePixelType;
  typedef typename Superclass::MovingImageType         MovingImageType;
  typedef typename Superclass::MovingImageType::PixelType MovingImagePixelType;
  typedef typename Superclass::FixedImageConstPointer
   FixedImageConstPointerType;
  typedef typename Superclass::MovingImageConstPointer
   MovingImageConstPointerType;


  itkStaticConstMacro(FixedImageDimension, unsigned int, TFixedImage::ImageDimension);

  /** Types for transforming the fixed image */
  typedef itk::Image< FixedImagePixelType,
                itkGetStaticConstMacro( FixedImageDimension ) >
              TransformedMovingImageType;

  typedef itk::ResampleImageFilter< MovingImageType, TransformedMovingImageType >
                              TransformMovingImageFilterType;

  typedef itk::Image< RealType,
                itkGetStaticConstMacro( FixedImageDimension ) >
              FixedGradientImageType;

  typedef typename FixedGradientImageType::ConstPointer      FixedGradientImageConstPointerType;



  typedef itk::CastImageFilter< FixedImageType, FixedImageType >
                              CastFixedImageFilterType;
  typedef typename CastFixedImageFilterType::Pointer CastFixedImageFilterPointer;

  typedef itk::CastImageFilter< FixedImageType, FixedGradientImageType >
                              CastFixedGradientImageFilterType;
  typedef typename CastFixedGradientImageFilterType::Pointer CastFixedGradientImageFilterPointer;
typedef  itk::GradientMagnitudeRecursiveGaussianImageFilter<
                    FixedGradientImageType,
                    FixedGradientImageType >  GradientFixedImageFilterType;
typedef typename GradientFixedImageFilterType::Pointer GradientFixedImageFilterTypePointer;

typedef   itk::RescaleIntensityImageFilter<
                    FixedGradientImageType,
                    FixedGradientImageType >  RescaleGradientFixedImageFilterType;
typedef typename RescaleGradientFixedImageFilterType::Pointer RescaleGradientFixedImageFilterTypePointer;


typedef typename FixedGradientImageType::PixelType FixedGradientPixelType;


typedef   itk::RescaleIntensityImageFilter<
                    FixedImageType,
                    FixedImageType >  RescaleFixedImageFilterType;
typedef typename RescaleFixedImageFilterType::Pointer RescaleFixedImageFilterTypePointer;


/** Types for transforming the moving image */

itkStaticConstMacro( MovedImageDimension, unsigned int,
               MovingImageType::ImageDimension );

typedef itk::Image< RealType,
              itkGetStaticConstMacro( MovedImageDimension ) >
             MovedGradientImageType;

typedef typename MovedGradientImageType::ConstPointer      MovedGradientImageConstPointerType;

typedef typename MovingImageType::ConstPointer      MovedImageConstPointerType;


typedef itk::CastImageFilter< TransformedMovingImageType, MovedGradientImageType >
                                   CastMovedGradientImageFilterType;
typedef typename CastMovedGradientImageFilterType::Pointer CastMovedGradientImageFilterTypePointer;


typedef itk::CastImageFilter< TransformedMovingImageType, TransformedMovingImageType >
                                   CastMovedImageFilterType;
typedef typename CastMovedImageFilterType::Pointer CastMovedImageFilterTypePointer;



typedef   itk::GradientMagnitudeRecursiveGaussianImageFilter<
                    MovedGradientImageType,
                    MovedGradientImageType >  GradientMovedImageFilterType;


typedef typename GradientMovedImageFilterType::Pointer GradientMovedImageFilterTypePointer;

typedef   itk::RescaleIntensityImageFilter<
                    MovedGradientImageType,
                    MovedGradientImageType >  RescaleGradientMovedImageFilterType;

typedef typename RescaleGradientMovedImageFilterType::Pointer RescaleGradientMovedImageFilterTypePointer;


typedef   itk::RescaleIntensityImageFilter<
                    MovingImageType,
                    MovingImageType >  RescaleMovedImageFilterType;

typedef typename RescaleMovedImageFilterType::Pointer RescaleMovedImageFilterTypePointer;


typedef typename MovedGradientImageType::PixelType MovedGradientPixelType;
```

```
  /** Initializes the metric. */
  void Initialize() throw (ExceptionObject);

  /** Sets the step length used to calculate the derivative. */
  itkSetMacro( DerivativeStepLength, double );

  /** Returns the step length used to calculate the derivative. */
  itkGetMacro( DerivativeStepLength, double );

  /** The scales type. */
  typedef Array<double> ScalesType;

  /** Sets the derivative step length scales. */
  itkSetMacro( DerivativeStepLengthScales, ScalesType );

  /** Returns the derivate step length scales. */
  itkGetConstReferenceMacro(DerivativeStepLengthScales, ScalesType);

  /** Gets and sets alpha parameter. If alpha = 1, the JIG
   * would have a greater impact in the measure. If alpha = 0,
   * the JII would have a greater impact in the measure.
   * Default value is 1.0  */
  itkSetMacro(Alpha, double);
  itkGetMacro(Alpha, double);

  /** Gets and sets sigma parameter. Sigma controls the effect of
   * the edges (larger sigma implies larger blurring of edges)
   * Default value is 1.0  */
  itkSetMacro(Sigma, double);
  itkGetMacro(Sigma, double);

  /**  Get the value for single valued optimizers. */
  MeasureType GetValue(const TransformParametersType& parameters) const;

  /** Get the derivatives of the match measure. */
  void GetDerivative(const TransformParametersType & parameters,
               DerivativeType & derivative) const;

  /**  Get value and derivatives for multiple valued optimizers. */
  void GetValueAndDerivative(const TransformParametersType & parameters,
                    MeasureType& Value,
                    DerivativeType& Derivative) const;

  /** Write gradient images to a files for debugging purposes. */
  void WriteGradientImagesToFiles(void) const;


protected:
  /** Constructor is protected to ensure that \c New() function is used to
     create instances. */
  CombinedJaccardIndexIntensityAndGradientImageToImageMetric();
  virtual ~CombinedJaccardIndexIntensityAndGradientImageToImageMetric() {};


  /** PrintSelf funtion */
  void PrintSelf(std::ostream& os, Indent indent) const;


private:
  CombinedJaccardIndexIntensityAndGradientImageToImageMetric(const Self&); //purposely not implemented
  void operator=(const Self&); //purposely not implemented

  /** The step length used to calculate the derivative. */
  double m_DerivativeStepLength;

  /** The derivative step length scales. */
```

```
ScalesType m_DerivativeStepLengthScales;

 /** The filter for transforming the moving image. */
typename TransformMovingImageFilterType::Pointer m_TransformMovingImageFilter;

/** The gradients of the fixed image */
CastFixedGradientImageFilterPointer        m_CastFixedGradientImageFilter;
CastFixedImageFilterPointer        m_CastFixedImageFilter;

GradientFixedImageFilterTypePointer  m_GradientFixedImageFilter;

RescaleGradientFixedImageFilterTypePointer  m_RescaleGradientFixedImageFilter;

RescaleFixedImageFilterTypePointer   m_RescaleFixedImageFilter;

/** The gradients of the moving image */
CastMovedImageFilterTypePointer          m_CastMovedImageFilter;
CastMovedGradientImageFilterTypePointer  m_CastMovedGradientImageFilter;

GradientMovedImageFilterTypePointer  m_GradientMovedImageFilter;

RescaleGradientMovedImageFilterTypePointer  m_RescaleGradientMovedImageFilter;

RescaleMovedImageFilterTypePointer         m_RescaleMovedImageFilter;

/** A factor multiplied by JIG . */
double m_Alpha;

double m_Sigma;

};

} // end namespace itk

#ifndef ITK_MANUAL_INSTANTIATION
#include "itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric.txx"
#endif

#endif // __itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric_h


/*=========================================================================

 Program:   Insight Segmentation & Registration Toolkit
 Module:    $RCSfile: itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric.txx,v $
 Language:  C++
 Date:      $Date: 2007/08/09 20:03:00 $
 Version:   $Revision: 0.01 $

 Copyright (c) Lino Ramirez. All rights reserved.

    This software is distributed WITHOUT ANY WARRANTY; without even
    the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
    PURPOSE.  See the above copyright notices for more information.

=========================================================================*/
#ifndef __itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric_txx
#define __itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric_txx

#include "itkArray.h"
#include "itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric.h"
#include "itkNumericTraits.h"
#include "itkImageRegionConstIterator.h"
#include "itkImageRegionConstIteratorWithIndex.h"

#include <iostream>
#include <iomanip>
#include <stdio.h>
```

```
#include "itkSimpleFilterWatcher.h"


//----------------------------------------------------------------
namespace itk
{
template <class TFixedImage, class TMovingImage>
CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage,TMovingImage>
::CombinedJaccardIndexIntensityAndGradientImageToImageMetric()
{
 itkDebugMacro("Constructor");

 m_DerivativeStepLength = 0.1;
 m_DerivativeStepLengthScales.Fill(1);
 m_TransformMovingImageFilter = 0;
 m_Alpha = 1.0;
 m_Sigma = 1.0;

}


//----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage, TMovingImage>
::Initialize() throw (ExceptionObject)
{
 Superclass::Initialize();
 if (!this->m_FixedImage)
   {
   itkExceptionMacro(<<"Fixed image has not been set.");
   }
 else if (!this->m_MovingImage)
   {
   itkExceptionMacro(<<"Moving image has not been set.");
   }


 if ( ! this->GetComputeGradient() )
   {
   itkExceptionMacro(<<"Gradients must be calculated");
   }


 // Create the filter to resample the moving image

 m_TransformMovingImageFilter = TransformMovingImageFilterType::New();

 m_TransformMovingImageFilter->SetTransform(    this->m_Transform );
 m_TransformMovingImageFilter->SetInterpolator( this->m_Interpolator );
 m_TransformMovingImageFilter->SetInput( this->m_MovingImage );

 m_TransformMovingImageFilter->SetDefaultPixelValue( 0 );

 m_TransformMovingImageFilter->SetSize( this->m_FixedImage->GetLargestPossibleRegion().GetSize() );
 m_TransformMovingImageFilter->SetOutputOrigin( this->m_FixedImage->GetOrigin() );
 m_TransformMovingImageFilter->SetOutputSpacing( this->m_FixedImage->GetSpacing() );
 m_TransformMovingImageFilter->SetOutputDirection( this->m_FixedImage->GetDirection() );


 // Compute the image gradients
 // ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 // Compute the gradient of the fixed image


 m_CastFixedImageFilter = CastFixedImageFilterType::New();
 m_CastFixedImageFilter->SetInput( this->m_FixedImage );
```

```
 m_CastFixedGradientImageFilter = CastFixedGradientImageFilterType::New();
 m_CastFixedGradientImageFilter->SetInput( this->m_FixedImage );

 m_GradientFixedImageFilter = GradientFixedImageFilterType::New();
 m_GradientFixedImageFilter->SetInput( m_CastFixedGradientImageFilter->GetOutput() );
 m_GradientFixedImageFilter->SetSigma( m_Sigma  );

 m_GradientFixedImageFilter->UpdateLargestPossibleRegion();

 m_RescaleGradientFixedImageFilter  = RescaleGradientFixedImageFilterType::New();
 m_RescaleGradientFixedImageFilter->SetInput( m_GradientFixedImageFilter->GetOutput() );

 m_RescaleGradientFixedImageFilter->SetOutputMinimum(  0.0  );
 m_RescaleGradientFixedImageFilter->SetOutputMaximum(  1.0  );

 // scale intensity values of fixed image
 m_RescaleFixedImageFilter  = RescaleFixedImageFilterType::New();
 m_RescaleFixedImageFilter->SetInput( m_CastFixedImageFilter->GetOutput() );

 m_RescaleFixedImageFilter->SetOutputMinimum(  0.0  );
 m_RescaleFixedImageFilter->SetOutputMaximum(  1.0  );



 // Compute the gradient of the transformed moving image

 m_CastMovedGradientImageFilter = CastMovedGradientImageFilterType::New();
 m_CastMovedGradientImageFilter->SetInput( m_TransformMovingImageFilter->GetOutput() );

 m_GradientMovedImageFilter = GradientMovedImageFilterType::New();
 m_GradientMovedImageFilter->SetInput( m_CastMovedGradientImageFilter->GetOutput() );
 m_GradientMovedImageFilter->SetSigma(  m_Sigma  );

 m_GradientMovedImageFilter->UpdateLargestPossibleRegion();



 m_RescaleGradientMovedImageFilter  = RescaleGradientMovedImageFilterType::New();
 m_RescaleGradientMovedImageFilter->SetInput( m_GradientMovedImageFilter->GetOutput() );

 m_RescaleGradientMovedImageFilter->SetOutputMinimum(  0.0  );
 m_RescaleGradientMovedImageFilter->SetOutputMaximum(  1.0  );


 // scale intensity values of moving image

 m_CastMovedImageFilter = CastMovedImageFilterType::New();
 m_CastMovedImageFilter->SetInput( m_TransformMovingImageFilter->GetOutput() );

 m_RescaleMovedImageFilter  = RescaleMovedImageFilterType::New();
 m_RescaleMovedImageFilter->SetInput( m_CastMovedImageFilter->GetOutput() );

 m_RescaleMovedImageFilter->SetOutputMinimum(  0.0  );
 m_RescaleMovedImageFilter->SetOutputMaximum(  1.0  );


}


//----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
typename CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage,TMovingImage>::MeasureType
CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage,TMovingImage>
::GetValue(const TransformParametersType& parameters) const
{
  itkDebugMacro("GetValue( " << parameters << " ) ");
  //
```

```
// computing JIG
//

this->SetTransformParameters( parameters );
m_TransformMovingImageFilter->UpdateLargestPossibleRegion();

MeasureType JIG = NumericTraits < MeasureType > :: Zero;

  // Compute the JIG
  m_RescaleGradientFixedImageFilter->UpdateLargestPossibleRegion();
  m_RescaleGradientMovedImageFilter->UpdateLargestPossibleRegion();


  FixedGradientImageConstPointerType pGradientFixedImage = this->m_RescaleGradientFixedImageFilter->GetOutput();
  ImageRegionConstIterator<FixedGradientImageType> fixedGradientIterator(pGradientFixedImage,
                                   pGradientFixedImage->GetBufferedRegion());

  fixedGradientIterator.GoToBegin();

  MovedGradientImageConstPointerType pGradientMovedImage = this->m_RescaleGradientMovedImageFilter->GetOutput();
  ImageRegionConstIterator<MovedGradientImageType>  movedGradientIterator(pGradientMovedImage,
                                    pGradientMovedImage->GetBufferedRegion());
  movedGradientIterator.GoToBegin();


  this->m_NumberOfPixelsCounted = 0;

  typedef  typename NumericTraits< MeasureType >::AccumulateType AccumulateType;

  AccumulateType sng  = NumericTraits< AccumulateType >::Zero;
  AccumulateType sdg  = NumericTraits< AccumulateType >::Zero;

  while (!fixedGradientIterator.IsAtEnd() )
   {

   //Get the moving and fixed image gradients
   RealType movingValue  = movedGradientIterator.Value();
   RealType fixedValue   = fixedGradientIterator.Value();


   // sn += min(fixedValue, movingValue);
   // sd += max(fixedValue, movingValue);
   if (fixedValue < movingValue)
    {
    sng += fixedValue;
    sdg += movingValue;
    }
   else
    {
    sng += movingValue;
    sdg += fixedValue;
    }

   this->m_NumberOfPixelsCounted++;
   ++fixedGradientIterator;
   ++movedGradientIterator;
  }

  if( this->m_NumberOfPixelsCounted > 0 && sdg != 0.0)
   {
   JIG = sng / sdg;
   }

//
// computing JII
//

MeasureType JII = NumericTraits < MeasureType > :: Zero;
```

```
// Compute the JII
m_RescaleFixedImageFilter->UpdateLargestPossibleRegion();
m_RescaleMovedImageFilter->UpdateLargestPossibleRegion();


FixedImageConstPointerType pFixedImage = this->m_RescaleFixedImageFilter->GetOutput();
ImageRegionConstIterator<FixedImageType> fixedIterator(pFixedImage,
                                  pFixedImage->GetBufferedRegion());

fixedIterator.GoToBegin();

MovedImageConstPointerType pMovedImage = this->m_RescaleMovedImageFilter->GetOutput();
ImageRegionConstIterator<MovingImageType>  movedIterator(pMovedImage,
                                   pMovedImage->GetBufferedRegion());
movedIterator.GoToBegin();


this->m_NumberOfPixelsCounted = 0;

AccumulateType sn  = NumericTraits< AccumulateType >::Zero;
AccumulateType sd  = NumericTraits< AccumulateType >::Zero;

while (!fixedIterator.IsAtEnd() )
  {

  //Get the moving and fixed image intensitry values
  RealType movingValue  = movedIterator.Value();
  RealType fixedValue   = fixedIterator.Value();


  // sn += min(fixedValue, movingValue);
  // sd += max(fixedValue, movingValue);
  if (fixedValue < movingValue)
    {
    sn += fixedValue;
    sd += movingValue;
    }
  else
    {
    sn += movingValue;
    sd += fixedValue;
    }

  this->m_NumberOfPixelsCounted++;
  ++fixedIterator;
  ++movedIterator;
 }

if( this->m_NumberOfPixelsCounted > 0 && sd != 0.0)
  {
  JII = sn / sd;
  }


//computing similarity measure
MeasureType measure = NumericTraits < MeasureType > :: Zero;

measure = m_Alpha * JIG + (1.0 - m_Alpha) * JII;


return ( -1.0 * measure );

}


//-----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
```

```
void
CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage,TMovingImage>
::GetDerivative(const TransformParametersType& parameters,
            DerivativeType& derivative) const
{
 itkDebugMacro("GetDerivative( " << parameters << " ) ");

 const double delta = 0.001;
 TransformParametersType testPoint;
 testPoint = parameters;

 const unsigned int numberOfParameters = this->GetNumberOfParameters();
 derivative = DerivativeType( numberOfParameters );

 for( unsigned int i=0; i<numberOfParameters; i++)
   {
   testPoint[i] -= delta;
   const MeasureType valuep0 = this->GetValue( testPoint );
   testPoint[i] += 2*delta;
   const MeasureType valuep1 = this->GetValue( testPoint );
   derivative[i] = (valuep1 - valuep0 ) / ( 2 * delta );
   testPoint[i] = parameters[i];
   }
}


//--------------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void
CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage,TMovingImage>
::GetValueAndDerivative(const TransformParametersType& parameters,
                  MeasureType& value,
                  DerivativeType& derivative) const
{
  value = GetValue(parameters);
  this->GetDerivative(parameters, derivative);
}


//--------------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void
CombinedJaccardIndexIntensityAndGradientImageToImageMetric<TFixedImage,TMovingImage>
::PrintSelf(std::ostream& os, Indent indent) const
{
  Superclass::PrintSelf(os,indent);
  os << indent << "Derivative step length: " << m_DerivativeStepLength
    << std::endl;
  os << indent << "Derivative step length scales: ";
  os << m_DerivativeStepLengthScales << std::endl;
  os << indent << "sigma: " << m_Sigma << std::endl;
  os << indent << "alpha: " << m_Alpha << std::endl;
}


} // end namespace itk

#endif // itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric_txx
```

Appendix 3: Characterization of the Proposed Similarity Measure

Algorithm for characterizing the proposed similarity measure

Given	Two images A and B
Defined	x_{min}, the minimum translation in the x axis. x_{max}, the maximum translation in the x axis. y_{min}, the minimum translation in the y axis. y_{max}, the maximum translation in the y axis. α, the weighting factor σ, the smoothing factor
Processing	For each translation in the x axis ($x \in [x_{min}, x_{max}]$) For each translation in the y axis ($y \in [y_{min}, y_{max}]$) a. Transform image B with the given translations b. Apply interpolation to find the pixel values at the requested positions c. Calculate the combined fuzzy Jaccard index by using: $CFJI(A,B) = \alpha \cdot FJI_g(A,B) + (1-\alpha) \cdot FJI_i(A,B)$ d. Output the *CFJI* value
Result	The *CFJI* values for a given set of translations

Source code for characterizing the proposed similarity measure

```
/*=========================================================================

  Program:   Insight Segmentation & Registration Toolkit
  Module:    $RCSfile: SimilarityMeasure.cxx,v $
  Language:  C++
  Date:      $Date: 2007/08/10 08:34:00 $
  Version:   $Revision: 0.1 $

  Copyright (c) Lino Ramirez. All rights reserved.

     This software is distributed WITHOUT ANY WARRANTY; without even
     the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
     PURPOSE.  See the above copyright notices for more information.

=========================================================================*/
#if defined(_MSC_VER)
#pragma warning ( disable : 4786 )
#endif

#include "itkImage.h"
#include "itkImageFileReader.h"
#include "itkImageFileWriter.h"

#include "itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric.h"
#include "itkTranslationTransform.h"
#include "itkNearestNeighborInterpolateImageFunction.h"

int main( int argc, char * argv[] )
{
  if( argc < 4 )
    {
    std::cerr << "Usage: " << std::endl;
    std::cerr << argv[0] << "  fixedImage  movingImage alpha" << std::endl;
    return 1;
    }

  const    unsigned int   Dimension = 2;
  typedef   float  PixelType;

  typedef itk::Image< PixelType, Dimension >   ImageType;

  typedef itk::ImageFileReader< ImageType >  ReaderType;

  ReaderType::Pointer fixedReader  = ReaderType::New();
```

```
ReaderType::Pointer movingReader = ReaderType::New();

fixedReader->SetFileName(  argv[ 1 ] );
movingReader->SetFileName( argv[ 2 ] );

try
 {
 fixedReader->Update();
 movingReader->Update();
 }
catch( itk::ExceptionObject & excep )
 {
 std::cerr << "Exception catched !" << std::endl;
 std::cerr << excep << std::endl;
 }

typedef itk::CombinedJaccardIndexIntensityAndGradientImageToImageMetric< ImageType, ImageType >  MetricType;
MetricType::Pointer metric = MetricType::New();
typedef itk::TranslationTransform< double, Dimension >  TransformType;

TransformType::Pointer transform = TransformType::New();


typedef itk::NearestNeighborInterpolateImageFunction<
                         ImageType, double >  InterpolatorType;

InterpolatorType::Pointer interpolator = InterpolatorType::New();

transform->SetIdentity();

ImageType::ConstPointer fixedImage  = fixedReader->GetOutput();
ImageType::ConstPointer movingImage = movingReader->GetOutput();

metric->SetTransform( transform );
metric->SetInterpolator( interpolator );

metric->SetFixedImage(  fixedImage  );
metric->SetMovingImage( movingImage );

metric->SetAlpha( atof(argv[ 3 ]) ); //valid only for JI
metric->SetSigma( 1.0 ); //valid only for JI

metric->SetFixedImageRegion(  fixedImage->GetBufferedRegion()  );

try
 {
```

```
 metric->Initialize();
 }
catch( itk::ExceptionObject & excep )
 {
 std::cerr << "Exception catched !" << std::endl;
 std::cerr << excep << std::endl;
 return -1;
 }

MetricType::TransformParametersType displacement( Dimension );

const int rangex = 10;
const int rangey = 10;
double dist;
MetricType::DerivativeType derivative;

for( int dx = -rangex; dx <= rangex; dx++ )
 {
 for( int dy = -rangey; dy <= rangey; dy++ )
  {
  displacement[0] = dx;
  displacement[1] = dy;
  dist = sqrt( dx* dx + dy*dy);
  const double value = metric->GetValue( displacement );
  metric->GetDerivative( displacement, derivative );
  std::cout << dx << "   " << dy << "   " << dist << " " << value << "   " <<std::endl;

  }
 }
return 0;
}
```

Appendix 4: Mean Squared Errors (MSE) Similarity Measure

Algorithm for computing the mean squared errors (*MSE*) similarity measure

Given	Two overlapping images A and B
Processing	For all the pixels in the region of overlap $j \in \{1,2,\ldots,N\}$ a. Calculate the *MSE* by: $MSE(A,B) = \frac{1}{N}\sum_{j=1}^{N}\left\|A(x_j) - B(x_j)\right\|^2$
Result	The Mean Squared Errors similarity measure *MSE*

Source code for computing the mean squared errors (*MSE*) similarity measure

```
/*=========================================================================

  Program:   Insight Segmentation & Registration Toolkit
  Module:    $RCSfile: itkMSEIntensityImageToImageMetric.h,v $
  Language:  C++
  Date:      $Date: 2007/08/18 01:50:10 $
  Version:   $Revision: 0.01 $

  Copyright (c) Lino Ramirez. All rights reserved.

     This software is distributed WITHOUT ANY WARRANTY; without even
     the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
     PURPOSE.  See the above copyright notices for more information.

=========================================================================*/
#ifndef __itkMSEIntensityImageToImageMetric_h
#define __itkMSEIntensityImageToImageMetric_h

#include "itkImageToImageMetric.h"

#include "itkNeighborhoodOperatorImageFilter.h"
#include "itkPoint.h"
#include "itkResampleImageFilter.h"

#include "itkCovariantVector.h"

namespace itk
{
 /** \class MSEIntensityImageToImageMetric
   \brief Computes a MSE from Intensity
    information between two images to be registered.

    The type of Intensity Information implemented in this class
    is given by the equation MSEI

    \f[  MSE(I(A),I(B)) \f]

    Where \$ I(A) \$ is the Intensity of image \$ A \$, rescaled to be in [0,1]
         \$ I(B) \$ is the Intensity of image \$ B \$, rescaled to be in [0,1]
   This class is templated over the type of the fixed and moving
   images to be compared.

 \ingroup RegistrationMetrics */
template <class TFixedImage, class TMovingImage>
class ITK_EXPORT MSEIntensityImageToImageMetric :
public ImageToImageMetric<TFixedImage, TMovingImage>
{
public:
  /** Standard class typedefs. */
  typedef MSEIntensityImageToImageMetric Self;
  typedef ImageToImageMetric<TFixedImage, TMovingImage> Superclass;
  typedef SmartPointer<Self>                   Pointer;
  typedef SmartPointer<const Self>                  ConstPointer;

  /** Method for creation through the object factory. */
  itkNewMacro(Self);


  /** Run-time type information (and related methods). */
  itkTypeMacro(MSEIntensityImageToImageMetric, ImageToImageMetric);

  /** Types transferred from the base class */
  typedef typename Superclass::RealType                 RealType;
  typedef typename Superclass::TransformType             TransformType;
  typedef typename Superclass::TransformPointer          TransformPointer;
  typedef typename Superclass::TransformParametersType
    TransformParametersType;
  typedef typename Superclass::TransformJacobianType
    TransformJacobianType;
  typedef typename Superclass::GradientPixelType         GradientPixelType;
  typedef typename Superclass::InputPointType            InputPointType;
  typedef typename Superclass::OutputPointType           OutputPointType;
  typedef typename Superclass::MeasureType               MeasureType;
  typedef typename Superclass::DerivativeType            DerivativeType;
  typedef typename Superclass::FixedImageType             FixedImageType;
  typedef typename Superclass::FixedImageType::PixelType  FixedImagePixelType;
  typedef typename Superclass::MovingImageType            MovingImageType;
  typedef typename Superclass::MovingImageType::PixelType MovingImagePixelType;
  typedef typename Superclass::FixedImageConstPointer
    FixedImageConstPointerType;
  typedef typename Superclass::MovingImageConstPointer
    MovingImageConstPointerType;


  itkStaticConstMacro(FixedImageDimension, unsigned int, TFixedImage::ImageDimension);

  /** Types for transforming the fixed image */
  typedef itk::Image< FixedImagePixelType,
                itkGetStaticConstMacro( FixedImageDimension ) >
              TransformedMovingImageType;

  typedef itk::ResampleImageFilter< MovingImageType, TransformedMovingImageType >
                              TransformMovingImageFilterType;

  /** Types for transforming the moving image */

  itkStaticConstMacro( MovedImageDimension, unsigned int,
                MovingImageType::ImageDimension );

  typedef typename MovingImageType::ConstPointer      MovedImageConstPointerType;



  /** Initializes the metric. */
  void Initialize() throw (ExceptionObject);
/** Sets the step length used to calculate the derivative. */
itkSetMacro( DerivativeStepLength, double );

/** Returns the step length used to calculate the derivative. */
itkGetMacro( DerivativeStepLength, double );

/** The scales type. */
typedef Array<double> ScalesType;

/** Sets the derivative step length scales. */
itkSetMacro( DerivativeStepLengthScales, ScalesType );

/** Returns the derivate step length scales. */
itkGetConstReferenceMacro(DerivativeStepLengthScales, ScalesType);


/**  Get the value for single valued optimizers. */
MeasureType GetValue(const TransformParametersType& parameters) const;

/** Get the derivatives of the match measure. */
void GetDerivative(const TransformParametersType & parameters,
            DerivativeType & derivative) const;

/**  Get value and derivatives for multiple valued optimizers. */
void GetValueAndDerivative(const TransformParametersType & parameters,
                 MeasureType& Value,
                 DerivativeType& Derivative) const;


protected:
 /** Constructor is protected to ensure that \c New() function is used to
   create instances. */
 MSEIntensityImageToImageMetric();
 virtual ~MSEIntensityImageToImageMetric() {};


 /** PrintSelf funtion */
 void PrintSelf(std::ostream& os, Indent indent) const;


private:
 MSEIntensityImageToImageMetric(const Self&); //purposely not implemented
 void operator=(const Self&); //purposely not implemented

 /** The step length used to calculate the derivative. */
 double m_DerivativeStepLength;

 /** The derivative step length scales. */
 ScalesType m_DerivativeStepLengthScales;

  /** The filter for transforming the moving image. */
 typename TransformMovingImageFilterType::Pointer m_TransformMovingImageFilter;


};

} // end namespace itk

#ifndef ITK_MANUAL_INSTANTIATION
#include "itkMSEIntensityImageToImageMetric.txx"
#endif

#endif // __itkMSEIntensityImageToImageMetric_h


/*=========================================================================
  Program:   Insight Segmentation & Registration Toolkit
  Module:    $RCSfile: itkMSEIntensityImageToImageMetric.txx,v $
  Language:  C++
  Date:      $Date: 2007/08/17 14:45:00 $
  Version:   $Revision: 0.01 $

  Copyright (c) Lino Ramirez. All rights reserved.

     This software is distributed WITHOUT ANY WARRANTY; without even
     the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
     PURPOSE.  See the above copyright notices for more information.

=========================================================================*/
#ifndef __itkMSEIntensityImageToImageMetric_txx
#define __itkMSEIntensityImageToImageMetric_txx

#include "itkArray.h"
#include "itkMSEIntensityImageToImageMetric.h"
#include "itkNumericTraits.h"
#include "itkImageRegionConstIterator.h"
#include "itkImageRegionConstIteratorWithIndex.h"

#include <iostream>
#include <iomanip>
#include <stdio.h>

#include "itkSimpleFilterWatcher.h"


//----------------------------------------------------------------------------
namespace itk
{
template <class TFixedImage, class TMovingImage>
MSEIntensityImageToImageMetric<TFixedImage,TMovingImage>
::MSEIntensityImageToImageMetric()
{
  itkDebugMacro("Constructor");

  m_DerivativeStepLength = 0.1;
  m_DerivativeStepLengthScales.Fill(1);
  m_TransformMovingImageFilter = 0;
}


//----------------------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void MSEIntensityImageToImageMetric<TFixedImage, TMovingImage>
::Initialize() throw (ExceptionObject)
{
  Superclass::Initialize();
  if (!this->m_FixedImage)
    {
    itkExceptionMacro(<<"Fixed image has not been set.");
    }
  else if (!this->m_MovingImage)
    {
    itkExceptionMacro(<<"Moving image has not been set.");
    }


  if ( ! this->GetComputeGradient() )
    {
    itkExceptionMacro(<<"Gradients must be calculated");
    }


  // Create the filter to resample the moving image
```

```
 m_TransformMovingImageFilter = TransformMovingImageFilterType::New();

 m_TransformMovingImageFilter->SetTransform(   this->m_Transform );
 m_TransformMovingImageFilter->SetInterpolator( this->m_Interpolator );
 m_TransformMovingImageFilter->SetInput( this->m_MovingImage );

 m_TransformMovingImageFilter->SetDefaultPixelValue( 0 );//

 m_TransformMovingImageFilter->SetSize( this->m_FixedImage->GetLargestPossibleRegion().GetSize() );
 m_TransformMovingImageFilter->SetOutputOrigin( this->m_FixedImage->GetOrigin() );
 m_TransformMovingImageFilter->SetOutputSpacing( this->m_FixedImage->GetSpacing() );
 m_TransformMovingImageFilter->SetOutputDirection( this->m_FixedImage->GetDirection() );

}

//-----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
typename MSEIntensityImageToImageMetric<TFixedImage,TMovingImage>::MeasureType
MSEIntensityImageToImageMetric<TFixedImage,TMovingImage>
::GetValue(const TransformParametersType& parameters) const
{
 itkDebugMacro("GetValue( " << parameters << " ) ");
 this->SetTransformParameters( parameters );
 m_TransformMovingImageFilter->UpdateLargestPossibleRegion();

 //
 // computing MSE
 //

 MeasureType MSE = NumericTraits < MeasureType > :: Zero;

  // Compute the MSE
  FixedImageConstPointerType pFixedImage = this->m_FixedImage;
  ImageRegionConstIterator<FixedImageType> fixedIterator(pFixedImage,
                                   pFixedImage->GetBufferedRegion());

  fixedIterator.GoToBegin();

  MovedImageConstPointerType pMovedImage = m_TransformMovingImageFilter->GetOutput();
  ImageRegionConstIterator<MovingImageType>  movedIterator(pMovedImage,
                                   pMovedImage->GetBufferedRegion());
  movedIterator.GoToBegin();



  this->m_NumberOfPixelsCounted = 0;

  typedef  typename NumericTraits< MeasureType >::AccumulateType AccumulateType;

  AccumulateType si  = NumericTraits< AccumulateType >::Zero;

  while (!fixedIterator.IsAtEnd() )
    {

    //Get the moving and fixed image intensitry values
    RealType movingValue  = movedIterator.Value();
    RealType fixedValue   = fixedIterator.Value();
    RealType diff         = movingValue - fixedValue;

    si += diff * diff;

    this->m_NumberOfPixelsCounted++;
    ++fixedIterator;
    ++movedIterator;
    }

  if( this->m_NumberOfPixelsCounted > 0 )
    {
```

```
    MSE = si / this->m_NumberOfPixelsCounted;
    }

  return ( MSE );

}

//----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void
MSEIntensityImageToImageMetric<TFixedImage,TMovingImage>
::GetDerivative(const TransformParametersType& parameters,
              DerivativeType& derivative) const
{
  itkDebugMacro("GetDerivative( " << parameters << " ) ");

  const double delta = 0.001;
  TransformParametersType testPoint;
  testPoint = parameters;

  const unsigned int numberOfParameters = this->GetNumberOfParameters();
  derivative = DerivativeType( numberOfParameters );

  for( unsigned int i=0; i<numberOfParameters; i++)
    {
    testPoint[i] -= delta;
    const MeasureType valuep0 = this->GetValue( testPoint );
    testPoint[i] += 2*delta;
    const MeasureType valuep1 = this->GetValue( testPoint );
    derivative[i] = (valuep1 - valuep0 ) / ( 2 * delta );
    testPoint[i] = parameters[i];
    }
}

//----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void
MSEIntensityImageToImageMetric<TFixedImage,TMovingImage>
::GetValueAndDerivative(const TransformParametersType& parameters,
                    MeasureType& value,
                    DerivativeType& derivative) const
{
  value = GetValue(parameters);
  this->GetDerivative(parameters, derivative);
}


//----------------------------------------------------------------
template <class TFixedImage, class TMovingImage>
void
MSEIntensityImageToImageMetric<TFixedImage,TMovingImage>
::PrintSelf(std::ostream& os, Indent indent) const
{
  Superclass::PrintSelf(os,indent);
}

} // end namespace itk

#endif // itkMSEIntensityImageToImageMetric_txx
```

Appendix 5: Model-to-Image Registration

Algorithm for performing model-to-image registration

Given	Two overlapping images *A* and *B*
Defined	α, the weighting factor σ, the smoothing factor Initial estimates for the geometrical transformation Convergence criterion for the optimization Maximum number of optimization iterations Any other parameter needed for the registration algorithm at hand.
Initialization	Initialize the geometrical transformation with the initial estimates Initialize the similarity measure
Processing	Find the rotation and translations that optimize the similarity measure following these steps: a. Transform image *B* with the given translations and rotations b. Apply interpolation to find the pixel values at the requested positions c. Calculate the similarity measure d. Adjust the transformation parameters to improve the similarity measure e. Repeat steps a. through d. until either the convergence criterion or the maximum number of iterations is reached
Result	The rotation and translations

Source code for performing model-to-image registration

```
/*=========================================================================

  Language:  C++
  Date:      $Date: 2007/09/17 18:13:00 $
  Version:   $Revision: 0.05 $

  Copyright (c) Lino Ramirez. All rights reserved.

     This software is distributed WITHOUT ANY WARRANTY; without even
     the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
     PURPOSE.  See the above copyright notices for more information.

=========================================================================*/
#if defined(_MSC_VER)
#pragma warning ( disable : 4786 )
#endif

#include "itkImageRegistrationMethod.h"
#include "itkCombinedJaccardIndexIntensityAndGradientImageToImageMetric.h"
#include "itkMSEIntensityImageToImageMetric.h"
```

```
#include "itkLinearInterpolateImageFunction.h"
#include "itkRegularStepGradientDescentOptimizer.h"
#include "itkImage.h"

#include "itkCenteredTransformInitializer.h"
#include "itkCenteredRigid2DTransform.h"

#include "itkImageFileReader.h"
#include "itkImageFileWriter.h"

#include "itkResampleImageFilter.h"
#include "itkCastImageFilter.h"
#include "itkSubtractImageFilter.h"
#include "itkRescaleIntensityImageFilter.h"
#include "itkIdentityTransform.h"


// The following section of code implements a Command observer
// that will monitor the evolution of the registration process.
//
#include "itkCommand.h"
class CommandIterationUpdate : public itk::Command
{
public:
  typedef CommandIterationUpdate   Self;
  typedef itk::Command             Superclass;
  typedef itk::SmartPointer<Self>  Pointer;
  itkNewMacro( Self );
protected:
  CommandIterationUpdate() {};
public:
  typedef itk::RegularStepGradientDescentOptimizer     OptimizerType;
  typedef   const OptimizerType   *    OptimizerPointer;

  void Execute(itk::Object *caller, const itk::EventObject & event)
    {
      Execute( (const itk::Object *)caller, event);
    }

  void Execute(const itk::Object * object, const itk::EventObject & event)
    {
      OptimizerPointer optimizer =
        dynamic_cast< OptimizerPointer >( object );
      if( ! itk::IterationEvent().CheckEvent( &event ) )
        {
        return;
        }
      std::cout << optimizer->GetCurrentIteration() << "   ";
      std::cout << optimizer->GetValue() << "   ";
      std::cout << optimizer->GetCurrentPosition() << std::endl;
    }
};


int main( int argc, char *argv[] )
{
  if( argc < 4 )
    {
    std::cerr << "Missing Parameters " << std::endl;
    std::cerr << "Usage: " << argv[0];
    std::cerr << " fixedImageFile  movingImageFile ";
    std::cerr << " outputImagefile ";
    std::cerr << " [differenceBeforeRegistration] ";
    std::cerr << " [differenceAfterRegistration] ";
    std::cerr << " [alpha] [sigma] ";
    std::cerr << " [steplength] ";
    std::cerr << " [metric: 1-CJI, 2-MSE] ";
  std::cerr << " [initialAngle] ";
  std::cerr << std::endl;
  return 1;
  }

const   unsigned int   Dimension = 2;
typedef  float          PixelType;

typedef itk::Image< PixelType, Dimension >  FixedImageType;
typedef itk::Image< PixelType, Dimension >  MovingImageType;

typedef itk::CenteredRigid2DTransform< double > TransformType;

typedef itk::RegularStepGradientDescentOptimizer       OptimizerType;


typedef itk::CombinedJaccardIndexIntensityAndGradientImageToImageMetric
                < FixedImageType, MovingImageType >  CJIMetricType;

typedef itk::MSEIntensityImageToImageMetric
                < FixedImageType, MovingImageType >  MSEMetricType;

typedef itk:: LinearInterpolateImageFunction< MovingImageType, double >
 InterpolatorType;
typedef itk::ImageRegistrationMethod< FixedImageType, MovingImageType >
 RegistrationType;

CJIMetricType::Pointer      cjiMetric        = CJIMetricType::New();
MSEMetricType::Pointer      mseMetric        = MSEMetricType::New();

OptimizerType::Pointer      optimizer     = OptimizerType::New();
InterpolatorType::Pointer   interpolator  = InterpolatorType::New();
RegistrationType::Pointer   registration  = RegistrationType::New();


unsigned int metric = 1;

if( argc > 9 )
  {
  metric =  atoi( argv[9] );
  }

if ( metric == 1 )
  {
  registration->SetMetric(        cjiMetric        );
  }

if ( metric == 2 )
  {
  registration->SetMetric(        mseMetric        );
  }

registration->SetOptimizer(     optimizer     );
registration->SetInterpolator(  interpolator  );

TransformType::Pointer  transform = TransformType::New();
registration->SetTransform( transform );

typedef itk::ImageFileReader< FixedImageType  > FixedImageReaderType;
typedef itk::ImageFileReader< MovingImageType > MovingImageReaderType;

FixedImageReaderType::Pointer  fixedImageReader  = FixedImageReaderType::New();
MovingImageReaderType::Pointer movingImageReader = MovingImageReaderType::New();

fixedImageReader->SetFileName(  argv[1] );
movingImageReader->SetFileName( argv[2] );

registration->SetFixedImage(    fixedImageReader->GetOutput()    );
```

```
registration->SetMovingImage(   movingImageReader->GetOutput()   );
fixedImageReader->Update();

registration->SetFixedImageRegion(
   fixedImageReader->GetOutput()->GetBufferedRegion() );

typedef itk::CenteredTransformInitializer<
                        TransformType,
                        FixedImageType,
                        MovingImageType >  TransformInitializerType;

TransformInitializerType::Pointer initializer = TransformInitializerType::New();

initializer->SetTransform(   transform );

initializer->SetFixedImage(  fixedImageReader->GetOutput() );
initializer->SetMovingImage( movingImageReader->GetOutput() );

initializer->GeometryOn();

initializer->InitializeTransform();

double initialAngle = 0.0;

if( argc > 10 )
  {
  initialAngle =  atof( argv[10] );
  }

transform->SetAngle( initialAngle );

registration->SetInitialTransformParameters( transform->GetParameters() );

typedef OptimizerType::ScalesType       OptimizerScalesType;
OptimizerScalesType optimizerScales( transform->GetNumberOfParameters() );
const double translationScale = 1.0 / 1000.0;

optimizerScales[0] =  1.0;
optimizerScales[1] =  translationScale;
optimizerScales[2] =  translationScale;
optimizerScales[3] =  translationScale;
optimizerScales[4] =  translationScale;

optimizer->SetScales( optimizerScales );

double steplength = 1.0;

if( argc > 8 )
  {
  steplength = atof( argv[8] );
  }

optimizer->SetMaximumStepLength( steplength );
optimizer->SetMinimumStepLength( 0.0001 );
optimizer->SetNumberOfIterations( 200 );

CommandIterationUpdate::Pointer observer = CommandIterationUpdate::New();
optimizer->AddObserver( itk::IterationEvent(), observer );

double alpha = 0.5;

if( argc > 6 )
  {
  alpha =  atof( argv[6] );
  }

double sigma = 1.0;
```

```
if( argc > 7 )
  {
  sigma =  atof( argv[7] );
  }

if ( metric == 1 )
  {
  cjiMetric->SetAlpha( alpha );
  cjiMetric->SetSigma( sigma );
  }

try
  {
  registration->StartRegistration();
  }
catch( itk::ExceptionObject & err )
  {
  std::cerr << "ExceptionObject caught !" << std::endl;
  std::cerr << err << std::endl;
  return -1;
  }

OptimizerType::ParametersType finalParameters =
            registration->GetLastTransformParameters();

const double finalAngle          = finalParameters[0];
const double finalRotationCenterX = finalParameters[1];
const double finalRotationCenterY = finalParameters[2];
const double finalTranslationX    = finalParameters[3];
const double finalTranslationY    = finalParameters[4];

const unsigned int numberOfIterations = optimizer->GetCurrentIteration();

const double bestValue = optimizer->GetValue();

const double finalAngleInDegrees = finalAngle * 45.0 / atan(1.0);

std::cout << std::endl;
std::cout << "Result = " << std::endl;
std::cout << " Angle (radians) " << finalAngle  << std::endl;
std::cout << " Angle (degrees) " << finalAngleInDegrees  << std::endl;
std::cout << " Center X      = " << finalRotationCenterX  << std::endl;
std::cout << " Center Y      = " << finalRotationCenterY  << std::endl;
std::cout << " Translation X = " << finalTranslationX  << std::endl;
std::cout << " Translation Y = " << finalTranslationY  << std::endl;
std::cout << " Iterations    = " << numberOfIterations << std::endl;
std::cout << " Metric value  = " << bestValue          << std::endl;

typedef itk::ResampleImageFilter< MovingImageType,
                        FixedImageType > ResampleFilterType;

TransformType::Pointer finalTransform = TransformType::New();

finalTransform->SetParameters( finalParameters );

ResampleFilterType::Pointer resampler = ResampleFilterType::New();

resampler->SetTransform( finalTransform );
resampler->SetInput( movingImageReader->GetOutput() );

FixedImageType::Pointer fixedImage = fixedImageReader->GetOutput();

resampler->SetSize(   fixedImage->GetLargestPossibleRegion().GetSize() );
resampler->SetOutputOrigin(  fixedImage->GetOrigin() );
resampler->SetOutputSpacing( fixedImage->GetSpacing() );
resampler->SetDefaultPixelValue( 100 );

typedef  unsigned char  OutputPixelType;
typedef itk::Image< OutputPixelType, Dimension > OutputImageType;

typedef itk::CastImageFilter< FixedImageType, OutputImageType >
 CastFilterType;

typedef itk::ImageFileWriter< OutputImageType >  WriterType;

WriterType::Pointer      writer =  WriterType::New();
CastFilterType::Pointer  caster =  CastFilterType::New();

writer->SetFileName( argv[3] );

caster->SetInput( resampler->GetOutput() );
writer->SetInput( caster->GetOutput()   );
writer->Update();

typedef itk::SubtractImageFilter<
                        FixedImageType,
                        FixedImageType,
                        FixedImageType > DifferenceFilterType;

DifferenceFilterType::Pointer difference = DifferenceFilterType::New();


typedef itk::RescaleIntensityImageFilter<
                        FixedImageType,
                        OutputImageType >   RescalerType;

RescalerType::Pointer intensityRescaler = RescalerType::New();

intensityRescaler->SetInput( difference->GetOutput() );
intensityRescaler->SetOutputMinimum(   0 );
intensityRescaler->SetOutputMaximum( 255 );

difference->SetInput1( fixedImageReader->GetOutput() );
difference->SetInput2( resampler->GetOutput() );

resampler->SetDefaultPixelValue( 1 );

WriterType::Pointer writer2 = WriterType::New();
writer2->SetInput( intensityRescaler->GetOutput() );

// Compute the difference image between the
// fixed and resampled moving image.
if( argc > 5 )
  {
  writer2->SetFileName( argv[5] );
  writer2->Update();
  }

typedef itk::IdentityTransform< double, Dimension > IdentityTransformType;
IdentityTransformType::Pointer identity = IdentityTransformType::New();

// Compute the difference image between the
// fixed and moving image before registration.
if( argc > 4 )
  {
  resampler->SetTransform( identity );
  writer2->SetFileName( argv[4] );
  writer2->Update();
  }

return 0;
}
```

Appendix 6: Computation of Measures of Overlap

Algorithm for computing the proposed overlap measures

Given	An image A and a Mask B
Defined	σ, the smoothing factor
Initialization	Calculate the gradient magnitudes for image A ($\nabla_{\sigma}(A)$)
Processing	For all the pixels in the images $j \in \{1,2,\ldots,N\}$ a. Calculate the *FJIG* by: $FJIG(A,B)=\frac{\sum_{j=1}^{N} min(\nabla_{\sigma}(A(x_j)), B(x_j))}{\sum_{j=1}^{N} max(\nabla_{\sigma}(A(x_j)), B(x_j))}$ For all the pixels in image A that correspond to the pixels in B that have the highest possible brightness level $j \in \{1,2,\ldots,M\}$ a. Calculate the *MeanG* by: $MeanG(A)=\frac{1}{M}\sum_{j=1}^{M} A(x_j)$ b. Calculate the *RangeG* by: $RangeG(A)=max(A(x))-min(A(x))$ c. Calculate the *StDevG* by: $StDevG(A)=\sqrt{\left(\frac{1}{M}\sum_{j=1}^{M}(A(x_j)-MeanG(A))^2\right)}$
Result	The overlap measures *FJIG*, *MeanG*, *RangeG*, and *StDevG*

Source code for computing the proposed overlap measures

```
#!/usr/bin/perl
# ==========================================================
# test_quality_assessment
#
#     Author:    Lino Ramirez
#     Date:      2007-October-22
#     Version:   0.5
# ==========================================================
use warnings;
use strict;
use PDL;
use PDL::Graphics::PGPLOT::Window;
use PDL::NiceSlice;
use PDL::IO::Pic;

# compute_FJIG($mask, $ROI) returns the FJIG
# for fuzzy set representations of a $mask and an $ROI
#
sub compute_FJIG {
   my ($mask, $ROI) = @_;

   my $flat_mask  = $mask->copy->flat;
   my $flat_roi   = $ROI->copy->flat;

   my $join = cat( $flat_mask, $flat_roi);

   my $intersection = sum( minimum( $join->xchg(0,1) ) );
   my $union        = sum( maximum( $join->xchg(0,1) ) );

   my $JIG = 0;
   if ( $union > 0 ) {
      $JIG = $intersection / $union;
   }

   return $JIG;
}

# find_stats
sub find_stats{
   my ( $mask, $ROI ) = @_;
   my ( $ind_x, $ind_y ) = whichND( $mask == 255 );
   return stats( $ROI( $ind_x, $ind_y ) );
}

# ==========================================================
# main
# ==========================================================

while (defined(my $root = <DATA>)) {
   chomp ($root);
   my $number_of_vertebrae = <DATA>;
   chomp ($number_of_vertebrae);

   foreach (1..$number_of_vertebrae){
      my $vertebra = <DATA>;
      chomp ($vertebra);
      my $top_ROI_name = $root . $vertebra . "ROI_T.png";
      my $top_output_name  = $root . $vertebra . "output_T.png";

      my $input_mask = rim( $top_output_name );
      my $input_roi  = rim( $top_ROI_name );

      my $gradient_file = $top_ROI_name . "_gradient.png";
```

```
    my $sigma = 1.0; # <========================================= sigma
    system "./CreatingGradientImage $top_ROI_name $gradient_file $sigma";

    my $gradient_roi  = rim( $gradient_file );
    my $JI = compute_FJIG( $input_mask, $gradient_roi );
    my ($mean, $prms, $median, $min, $max, $adev, $rms) = find_stats( $input_mask, $gradient_roi );
    print "$JI\t$mean\t$prms\t$median\t$min\t$max\t$adev\t$rms\n\n";

    my $bot_ROI_name = $root . $vertebra . "ROI_B.png";
    my $bot_output_name  = $root . $vertebra . "output_B.png";
    $input_mask = rim( $bot_output_name );
    $input_roi  = rim( $bot_ROI_name );

    $gradient_file = $bot_ROI_name . "_gradient.png";

    $sigma = 1.0; # <========================================= sigma
    system "./CreatingGradientImage $bot_ROI_name $gradient_file $sigma";

    $gradient_roi  = rim( $gradient_file );

    $JI = compute_FJIG( $input_mask, $gradient_roi );
    ($mean, $prms, $median, $min, $max, $adev, $rms) = find_stats( $input_mask, $gradient_roi );
    print "$JI\t$mean\t$prms\t$median\t$min\t$max\t$adev\t$rms\n\n";

  }

}

__DATA__
Img0020R
2
L4
T9
```

Appendix 7: Open Source Software Contributions

During the realization of this work, four classes, one example and one application were contributed to the Insight Toolkit (ITK, http://www.itk.org). ITK is an open source set of libraries for medical image processing, segmentation, and registration. This toolkit started in 1999 with a US$ 10M investment by NIH (National Institute of Health) and collaborative work from 3 companies and 3 universities (with 6 additional universities working as subcontractors). Today, it is influencing the international medical imaging research community. User and developer mailing lists include over 1000 members in more than 30 countries. The set of applications using ITK include diverse areas such as: image guided intervention, longitudinal studies of Alzheimer's patients using MRI, and vascular segmentation. ITK is a clear success in the field of medical imaging. Contributions to this toolkit are developed following a state-of-the-art software development process that involves: high-intensity design, test, and implement cycle; support from web-enabled tools; and automated testing integrated with the software development. This approach guarantees that problems are identified and fixed immediately.

The aforementioned contributions to ITK represent the first (and only) effort from members from the University of Alberta to contribute to the Medical Image Computing Open Source Community. This community includes Universities and Companies such as: University of Utah, University of Pennsylvania, University of North Carolina at Chapel Hill, Columbia University, University of Pittsburgh, General Electric Corporate Research and Development, Insightful, and Kitware.

Classes: Morphological Image Processing and Arithmetic Operations

Class 1: Code/BasicFilters/itkGrayscaleMorphologicalOpeningImageFilter.h

Class 1: Code/BasicFilters/itkGrayscaleMorphologicalOpeningImageFilter.txx

Class 2: Code/BasicFilters/itkGrayscaleMorphologicalClosingImageFilter.h

Class 2: Code/BasicFilters/itkGrayscaleMorphologicalClosingImageFilter.txx

Class 3: Code/BasicFilters/itkConstrainedValueAdditionImageFilter.h

Class 4: Code/BasicFilters/itkConstrainedValueDifferenceImageFilter.h

Code available at:

http://www.itk.org/cgi-bin/viewcvs.cgi/Code/BasicFilters/?root=Insight

Example: Morphological Image Enhancement

Examples/Filtering/MorphologicalImageEnhancement.cxx

Code available at:

http://www.itk.org/cgi-bin/viewcvs.cgi/Examples/Filtering/?root=Insight

Application: 2-D Image Registration

ImageRegistration2D/CommandIterationUpdate.h

ImageRegistration2D/ImageRegistration2D.cxx

ImageRegistration2D/ImageRegistration2D.h

ImageRegistration2D/ImageRegistrationConsole.cxx

ImageRegistration2D/ImageRegistrationConsole.h

ImageRegistration2D/ImageRegistrationConsoleBase.cxx

ImageRegistration2D/ImageRegistrationConsoleBase.h

ImageRegistration2D/ImageRegistrationConsoleGUI.fl

Code available at:

http://www.itk.org/cgi-bin/viewcvs.cgi/ImageRegistration2D/?root=InsightApplications